Tietovarastojen Validiteetti

Matti Tahvanainen

Pro Gradu -tutkielma
HELSINGIN YLIOPISTO
Tietojenkäsittelytieteen laitos

Helsinki, 2. helmikuuta 2018

HELSINGIN YLIOPISTO — HELSINGFORS UNIVERSITET — UNIVERSITY OF HELSINKI

Tiedekunta — Fakultet — Faculty	Laitos — Institution — Department
Matemaattis-luonnontieteellinen	Tietojenkäsittelytieteen laitos

Tekijä — Författare — Author
Matti Tahvanainen

Työn nimi — Arbetets titel — Title

Tietovarastojen Validiteetti

Oppiaine — Läroämne — Subject
Tietojenkäsittelytiede

Työn laji — Arbetets art — Level	Aika — Datum — Month and year	Sivumäärä — Sidoantal — Number of pages
Pro Gradu -tutkielma	2. helmikuuta 2018	60

Tiivistelmä — Referat — Abstract

Yritysten päätöksenteko pohjautuu liiketoimintaa kuvaavaan aineistoon. Tätä aineistoa kerätään merkittäviä määriä ja se voi olla hajautettuna useaan eri järjestelmään, jotka eivät välttämättä ole yhteensopivia. Aineistoa voidaan kerätä tietovarastoihin (engl. data warehouse). Tietovarastot ovat suunniteltu säilömään valtavia määriä dataa, sekä muodostamaan automaattisesti erilaisia päätöksentekoa auttavia laskentoja, esimerkiksi automaattisia tuotteiden myyntiennusteita. Kun aineisto on kirjoitettu tietovarastoon, on mahdollista, että tietovarastoa käyttävä yritys näkee koko heidän liiketoimintaa kuvaavan aineiston ensimmäistä kertaa yhdessä järjestelmässä. Tämän aineiston tarkistus ja läpikäynti on tietovarastohankkeiden aikaavievin prosessi. Aineiston tarkistamis- ja läpikäymisprosessia kutsutaan usein datan validoinniksi (engl. data validation).

Tutkielma tehdään yritykselle, jonka päätuote on tietovarasto. Tutkielmassa toteutetaan datan validointia nopeuttava työkalu, joka etsii erilaisten validointisääntöjen avulla virheitä tietovarastoa käyttävän tahon liiketoimintaa kuvaavasta datasta. Datavirheellä tarkoitetaan esimerkiksi kirjoitusvirhettä tuotekoodissa. Työkalun kehitys aloitetaan tutustumalla aiheeseen liittyvään tutkimukseen, sekä keräämällä tietoa datan validointiin liittyvistä säännöistä ja käytänteistä case-yrityksessä työskenteleviltä asiantuntijoilta. Työkalun vaatimusmäärittely muodostetaan asiantuntijahaastattelujen, sekä löydetyn aineiston pohjalta. Työkalun pääasiallisena tarkoituksena on nopeuttaa tietovarastohankkeissa tapahtuvaa datan validointia. Nopeuttaminen tapahtuu automatisoimalla jokaisessa hankkeessa tapahtuvia datan validointiin liittyviä vaiheita tai sääntöjä.

Tutkielmassa tehtiin useita datan laatuun liittyviä havaintoja. Yritykset ja muut tahot keräävät huomattavasti dataa mallintaakseen omaa toimintaansa, jotta he pystyvät tekemään dataan perustuvia päätöksiä. Datan laatua on vaikeaa seurata, koska sitä kertyy lyhyessä ajassa merkittäviä määriä. Aiemmat tutkimukset osoittavat yritysten ja muiden tahojen menettävän huomattavia rahasummia datan heikon laadun seurauksena [31, 9]. Tämän tutkielman tulokset tukevat aiempia tutkimuksia. Tässä tutkielmassa kehitetty työkalu on löytänyt kaikista asiakkaille tehdyistä tietovarastoasennuksista datavirheitä tai -puutteita. Muut havainnot liittyvät ongelman ratkaisun teknisiin haasteisiin. Aiempaa aiheeseen liittyvää tutkimusta on jo vuosikymmenten takaa [18, 24], mutta varsinaista kattavaa automaattista ratkaisua ongelmaan ei ole. Kaupallisia datan laatua korjaavia ohjelmia on kymmeniä ellei satoja. Jokainen ohjelma lähestyy ongelmaa hieman eri tavalla. Tekninen yhteensopivuus tietovaraston, sekä irrallisen kaupallisen laadunvalvontaohjelman kanssa on vaikeaa muodostaa, sillä tietovarastot ovat usein suunniteltuja ratkaisemaan jotain rajattua ongelmaa. Erilaisia datan laatuun liittyviä sääntöjä on tuhansia ja suuri osa säännöistä on sidottuja datan sisältöön, joka tekee automatisoinnista haastavaa.

Tämä tutkielma osoittaa datan laadun olevan merkittävä ongelma tietovarastohankkeissa. Tutkielmassa muodostettiin määrittely, jonka pohjalta toteutettiin käytännön työkalu, joka etsii datavirheitä tietovarastosta. Toteutettu määrittely ja työkalu todettiin toimivaksi case-yrityksen tarpeisiin.

ACM Computing Classification System (CCS):
Information systems → Information storage systems
Information systems → Information systems applications
Software and its engineering→ Software creation and management

Avainsanat — Nyckelord — Keywords
Tietovarasto, datan laatu, datan validointi, datan puhdistaminen, ETL, automaattinen validointi

Säilytyspaikka — Förvaringsställe — Where deposited

Muita tietoja — Övriga uppgifter — Additional information

Sisältö

1 Johdanto ja tutkimuskysymykset

Yrityksen päätöksentekoa avustavat ja ohjaavat ohjelmistot ovat yleistyneet viime vuosikymmeninä. Yritysjohtajat priorisoivat liiketoimintatiedon hyödyntämisen (engl. business intelligence, BI) liittyvän tutkimuksen ja teknologian korkealle yrityksen kehittämissuunnitelmissa [44]. BI-työkalut ovat yrityksen liiketoimintaan liittyvän tiedon analysointiin tarkoitettuja ohjelmistoja, joilla voidaan esimerkiksi pyrkiä karsimaan tuotannon eri vaiheisiin kuluvaa aikaa, tai ennustaa tulevaisuuden näkymiä historiallisen myynnin perusteella. Analysointia ei kuitenkaan voida tehdä ilman yrityksen liiketoimintaan liittyvää aineistoa.

Analysointia voi olla tietyissä tapauksissa hankalaa toteuttaa aineiston keruun haasteiden vuoksi. Yrityksen liiketoimintatieto voi olla hajautettu moniin eri osiin. Liiketoimintatieto voi olla tallennettu esimerkiksi useampaan eri järjestelmään, jotka eivät ole toistensa kanssa yhteensopivia. Kansainvälisissä yrityksissä tieto voi olla tallessa useissa järjestelmissä eri mantereilla. Yrityksillä voi olla paljon dataa, mutta ei järjestelmää, joka yhdistää datan tai mahdollistaa sen tutkimisen useasta näkökulmasta [22]. Tietovarastot (engl. data warehouse) ovat suunniteltuja ratkaisemaan kyseiset ongelmat. Tietovarastoihin voidaan kerätä liiketoimintatietoa useasta eri lähteestä, jolloin yrityksen liiketoiminta-aineisto saadaan tallennettua yhteen paikkaan.

Tietovarastohankkeiden menestys riippuu sinne tallennetun datan laadusta [22, 5]. Datan laatu on avainasemassa päätöksenteossa ja erilaista automaattista laskentaa, kuten ennustelaskentaa tehdessä. Datan laadulla tarkoitetaan muun muassa sen uskottavuutta, relevanssia ja johdonmukaisuutta. Tietovarastohankkeissa aineistojen muokkaaminen laadukkaaksi on kallein ja aikaavievin osa [16]. Pääasiallinen syy datan käsittelyyn on eri lähteistä kerättyjen aineistojen yhteismitallistaminen vertailukelpoiseen muotoon. Niissä tapauksissa kun järjestelmän tietomalli noudattaa aina samaa perusmuotoa, on mahdollista muodostaa automaattinen tapa tunnistaa puutteita aineistojen eheydessä tai vertailukelpoisuudessa vaikka lähdejärjestelmät ovat heterogeenisiä.

Tässä tutkielmassa toteutetaan case-yritykselle työkalu, jonka avulla voidaan automaattisesti validoida lähdejärjestelmästä tietovarasto-ohjelmistoon saapuvaa dataa. Ohjelmisto laskee automaattisesti lähdejärjestelmästä saadun liiketoimintaa kuvastavan datan perusteella päätöksentekoa tukevaa ohjausdataa, joka tarkoittaa esimerkiksi tuotteiden tilausehdotuksia tai myyntiennusteita. Ohjausdataa muodostetaan asiakkaan liiketoimintaa kuvaavasta aineistosta. Jos datan laatu on heikkoa, ohjelmiston tuottama ohjausdata ei välttämättä tue asiakkaan päätöksentekoa. Tutkielmassa kehitettävä työkalu pyrkii löytämään datavirheitä tai -puutteita asiakkaan järjestelmistä eli lähdejärjestelmistä tulleesta datasta. Työkalun tarkoituksena on löytää ohjausdatan laadun kannalta kriittisiä datan laatuvirheitä tietovarastohankkeiden alkuvaiheessa. Tällöin datan laatua on mahdollista parantaa yhdessä asiak-

kaan kanssa normaalia aikaisemmassa vaiheissa, jolloin tietovarastohanke tuottaa asiakkaalle arvoa entistä aiemmin.

Tutkielman tarkoituksena on kartoittaa tietoa datan laadusta, sekä datan laadun vaikutuksista yrityksille, jotka käyttävät tietovarastoja päätöksentekoprosesseissaan. Datan laatuun liittyvästä kirjallisuudesta pyritään löytämään yleisiä tärkeitä laatutekijöitä. Näiden avulla datasta voidaan automaattisesti löytää puutteita. Kirjallisuuskatsauksessa etsitään keinoja datan validiteetin tarkistamiseen. Empiirisessä osiossa vastaavaa tehdään haastattelemalla case-yrityksen työntekijöitä, sekä seuraamaalla mitä asioita he etsivät datasta pyrkiessään varmistamaan sen oikeellisuutta eli validoimalla sitä. Tutkielman lopussa suunnitellaan ja toteutetaan validointityökalun prototyyppi, minkä tarkoituksena on tukea case-yrityksen asiantuntijoita datan validoinnissa. Seuraavat tutkimuskysymykset muodostettiin jäsentämään tutkielmaa:

1. Mitä teknisiä ja liiketoiminnallisia haittoja huonolaatuisesta datasta on tietovarastohankkeissa?

2. Kuinka tietovarastohankkeiden osapuolet tarkistavat datan validiteettia?

3. Onko mahdollista muodostaa tietovarastohankkeissa käytettävä automaatio, joka ilmoittaa datavirheistä, jotka ovat liiketoiminnallisesti haitallisia hankkeessa mukana oleville osapuolille?

Tutkielman luvuissa 2-4 paneudutaan aihealueeseen liittyvään teoriaan. Luvussa 2 esitellään tietovarasto ja siihen liittyvää terminologiaa. Luvussa 3 tutustutaan datan laatuun teknillisestä ja käyttäjän näkökulmasta. Luvussa 4 määritellään datan validointi ja validointiin liittyviä prosesseja sekä kerrotaan validointityökaluista. Luvut 5-8 ovat tutkielman empiirinen osa. Luvussa 5 esitellään yritys, jonne tutkimus tehdään ja kerrotaan tapaustutkimuksen toteutuksesta. Luvussa 6 kerrotaan, kuinka validointityökalu toteutetaan, kuvataan sen teknistä toimintaa mallien ja kaavioiden avulla ja esitellään sen muodostama raportti. Työkalua arvioidaan luvussa 7 vertaamalla sitä kirjallisuudessa esiteltyihin työkaluihin ja analysoimalla työkalun löydöksiä kolmessa asiakashankkeessa. Luvussa 8 pohditaan tutkielman löydöksiä, arvioidaan jatkokehitysmahdollisuuksia ja esitellään tutkielman rajoitteita.

2 Tietovarasto

Tässä luvussa tutustutaan tietovaraston käsitteeseen. Luvussa 2.1 käydään läpi tietovaraston määritelmä. Luvussa 2.2 esitellään tietovarastoon liittyvä keskeinen tiedonsiirtoprosessi, joka on niin kutsuttu ETL-prosessi (engl. extract, transform, load). Luvussa 2.3 kerrotaan tietovaraston hyödyistä.

2.1 Määritelmä

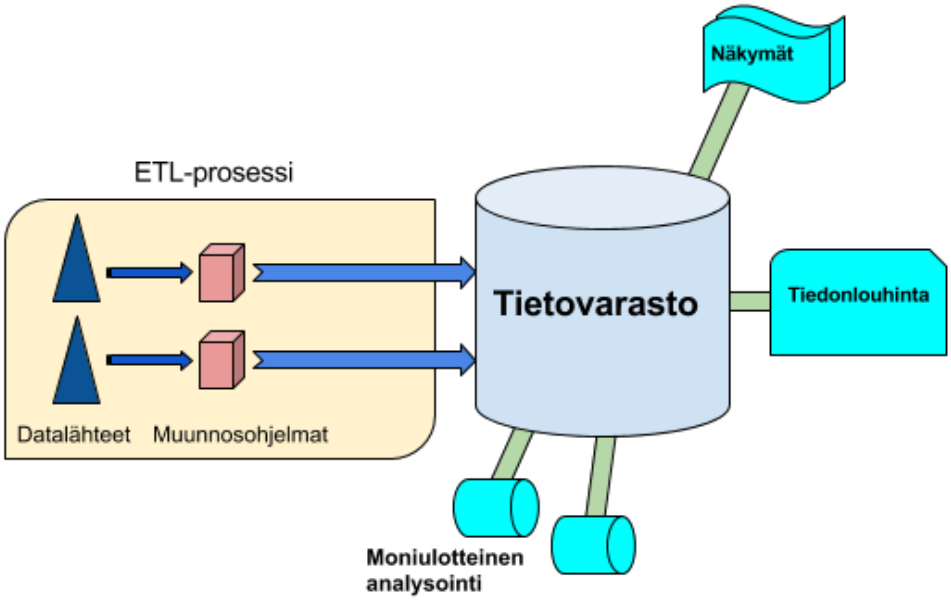
Dataan pohjautuva päätöksenteko on tullut suosioon yrityksissä viime vuosikymmeninä. On olemassa useita data-analyysiin tarkoitettuja työkaluja, niin sanottuja OLAP (online analytical processing) -järjestelmiä, jotka auttavat yritysten johtoa muodostamaan liiketoimintastrategioita ja pysymään kilpailussa mukana. Yritykset pitävät tärkeinä liiketoimintatiedon hallintaan keskittyviä teknologioita [13]. Liiketoimintatiedon hallinta tarkoittaa yrityksen muodostaman datan, kuten myyntitietojen, systemaattista keräämistä ja analysointia. Prosessiin voi kuulua datan kerääminen tietovarastoon, sen muokkaaminen ja analysointi sekä havainnollistaminen erilaisten mittareiden ja kuvaajien avulla.

Tietovarasto (engl. data warehouse) on järjestelmä, jonka tarkoituksena on tukea päätöksentekoa [19, 22]. Tietovarasto on yksi keino hallita liiketoimintatietoja. Tietovarastoon säilötään dataa ulkoisesta datalähteestä [19, 22]. Tietovarastohankkeita tehdään muun muassa eri alojen kaupallisille yrityksille, sillä niiden johtoryhmät haluavat pohjata päätökset ja strategiset linjaukset dataan [10]. Yritykset haluavat tietovarastohankkeilta muun muassa datan helpompaa saatavuutta ja mahdollisuutta nähdä data johdonmukaisessa esitysmuodossa. Tämän lisäksi tietovarastojen halutaan olevan mukautuvia ja joustavia muutoksille [22]. Tietovarastojen ajatellaan olevan tehokkain viimeisen kymmenen vuoden aikana nousseista liiketoiminnallista päätöksentekoa helpottavista teknologioista [1].

Kuvassa 1 on esitelty tietovarasto ja siihen liittyvää käsitteistöä. Kuvassa vasemmalla on *ETL-prosessi*, joka on tämän tutkielman kannalta yksi keskeisimmistä käsitteistä. ETL tulee sanoista extract, transform ja load, mitkä suomentuvat sanoiksi haku, muunnos ja kirjoitus. ETL on tiedonsiirtoprosessi, jossa data haetaan asiakkaan järjestelmästä, muunnetaan haluttuun muotoon ja lopuksi kirjoitetaan tietovarastoon. Prosessi selitetään yksityiskohtaisemmin luvussa 2.2. Lisäksi siitä puhutaan tutkielman soveltavassa osassa.

Tietovaraston avulla dataa voidaan tutkia näkymissä, jotka ovat kuvan 1 oikeassa yläkulmassa. Data haetaan näkymille tietokantakyselyiden avulla. Kuvassa oikealla on käsite *tiedonlouhinta*. Tietovarastoon kirjoitetaan paljon dataa. Jos tietovarastoon kirjoitetaan esimerkiksi tuotteiden myyntidataa usean vuoden ajalta, on mahdollista muodostaa trendejä ja tilastoja, joiden avulla voidaan arvioida minä vuodenaikoina tuotteilla on eniten kysyntää. Datan avulla voidaan lisäksi projisoida tulevaisuutta eli ennustaa tuotteiden kysyntää. Tietovarastojen avulla dataa voidaan tutkia eri näkökulmista (kuvassa *Moniulotteinen analysointi*). Tietovarastoa käyttävä yritys voi analysoida myyntiä esimerkiksi maa-, tuote- tai myymäläkohtaisesti.

Tietovarastojen avulla voidaan tukea yrityksen operatiivista toimintaa. Operatiivisella toiminnalla tarkoitetaan esimerkiksi tuotteiden tilaamista, tuotannonsuunnittelua tai inventaarion hallintaa. Operatiivista toimintaa



Kuva 1: Tietovarastoon liittyvää terminologiaa [12].

voidaan tehdä toiminnanohjausjärjestelmällä (engl. enterprise resource planning, jatkossa ERP) [41]. Esimerkiksi ohjelmistotalo SAP tarjoaa ERP-ohjelmistoja [34]. Data haetaan tietovarastoon operatiivisista järjestelmistä.

Tietovarastot ovat kohdeorientoituvia (engl. subject oriented), integroituja (engl. integrated), aikasidonnaisia (engl. time variant), vakaita (engl. non-volatile), pitkäaikaista dataa säilyttäviä kokonaisuuksia, jotka tukevat johtoryhmen päätöksentekoa [19, 39].

Kohdeorientoituvuudella tarkoitetaan tietovaraston käsittelevän yrittäjyyteen liittyviä yleisiä käsitteitä [19]. Tietovarastossa on valmiina tietokantatauluja, jotka liittyvät yleisiin liiketoimintamuotoihin, kuten esimerkiksi tavarantoimittaja, tuote ja toimipiste. Kohdeorientoituvien ohjelmistojen vastakohtana voidaan pitää operatiivisia ohjelmistoja. Kohdeorientoituvuuden seurauksena tietovarastoon päätyvän datan halutaan aina tukevan päätöksentekoa.

Integroituvuudella tarkoitetaan, että tietovaraston data on aina haettu yhdesta tai useammasta lähteestä ja data on muokattu tietovaraston konventioiden mukaiseksi [19]. Data vastaanotetaan valmiiksi muodossa, joka on tietovaraston konventioiden mukainen tai data muokataan sopivaan muotoon, ennen kuin se kirjoitetaan tietovarastoon. Kun data tulee lähdejärjestelmästä, eli tietovaraston käyttöönottavan yrityksen tietokannasta, siinä voi olla useita eri muotoja ilmaista päivämäärä tai sukupuoli. Tietovarastoon kirjoitetaan data aina yhdessä päivämääräformaatissa ja sukupuoli ilmaistaan yhdellä tavalla.

Tietovaraston data on *aikasidonnaista* eli data liittyy aina johonkin ajanhetkeen [19]. Tietovarastosta voidaan hakea yleensä useamman vuoden takaista dataa. Tietovaraston tietokantataulujen avainarvoina käytetään usein aikaan viittaavia arvoja, kuten päiviä, kuukausia tai vuosia.

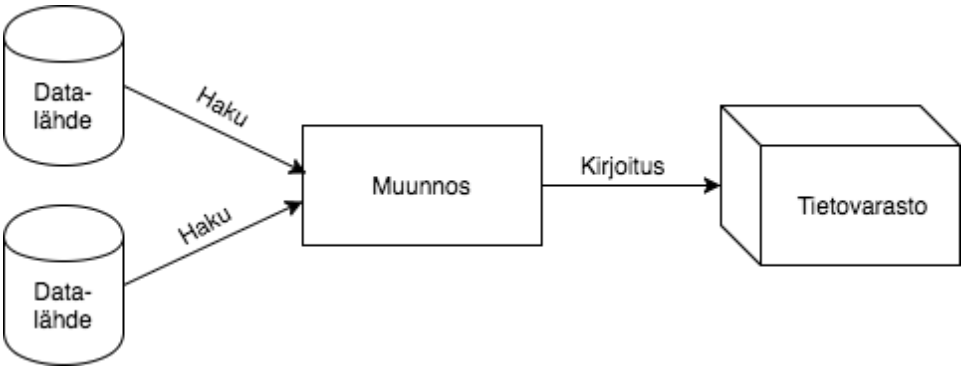
Yksi aikasidonnaisuuteen liittyvä ominaisuus tietovarastoissa on datan muuttumattomuus [19]. Kun data on kerran kirjoitettu tietovarastoon, sitä ei voi muuttaa. Data on on aina olemassa niin sanotuissa tilannekatsauksissa (engl. snapshot). Tietovarastoa voidaan ajatella yhtenä pitkänä sarjana tilannekatsauksia. Yksi tilannekatsaus kuvastaa tietovaraston tilaa tiettynä ajanhetkenä.

Operatiivisten työkalujen data reflektoi nykyhetkeä ja on ajantasaista [19]. Tietovaraston data integroidaan operatiivisesta järjestelmästä, jolloin se ei ole koskaan ajantasaista. Data kuvastaa operatiivisesta järjestelmästä viimeksi saatua tietoa. Operatiivisissa BI-työkaluissa aikajakso on merkittävästi pienempi kuin tietovarastoissa, sillä niiden käyttötarkoitukset liittyvät nykyajanhetkeen, ja niiden halutaan toimivan mahdollisimman tehokkaasti. Tämän seurauksena dataa ei yleensä ole saatavilla kuin joitakin kuukausia taaksepäin.

Vakaudella tarkoitetaan datan säilyvyyttä ja muuttumattomuutta [19]. Kun data on saatu kirjoitettua tietovarastoon ja tilannekatsaus on tehty, se ei enää muutu. Tietovarastoissa tapahtuu vain kahdenlaisia operaatiota: datan lukeminen sisään ja sen katseleminen.

2.2 Haku, Muunnos, Kirjoitus (Extract, Transform, Load)

Jotta data saadaan tietovarastoon, suoritetaan tyypillisesti kolmivaiheinen ETL-prosessi [30, 42, 10], joka tulee sanoista *extract*, *transform* ja *load* eli haku, muunnos ja kirjoitus. Yksinkertaistettu malli on esitelty kuvassa 2.



Kuva 2: Haku, Muunnos, Kirjoitus -malli yksinkertaistettuna

ETL-prosessin avulla siirretään dataa lähde- ja kohdejärjestelmän välillä. Toimivassa ETL-prosessissa data siirtyy lähdejärjestelmästä kohdejärjestelmään automaattisesti. ETL-prosessi on kuin tulkki, joka muodostaa kom-

munikointiyhteyden kahden eri kieliä puhuvan henkilön välillä. Prosessissa noudetaan lähdejärjestelmästä dataa ja muokataan se kohdejärjestelmälle soveltuvaan muotoon, jonka jälkeen data kirjoitetaan kohdejärjestelmään.

Hakuvaiheessa (engl. extract) tieto kerätään yhdestä tai useammasta lähdejärjestelmästä [10]. Hakuvaiheessa otetaan huomioon lähde- ja kohdejärjestelmien erot, kuten eri käyttöjärjestelmät, tietokannan hallintajärjestelmät tai eri kommunikointiprotokollat [10]. Hakuvaihe jaetaan kahteen eri osaan: alustavaan hakuun ja inkrementaaliseen, vähittäisesti kasvavaan hakuun [10]. Alustavassa haussa testataan tiedonsiirtoyhteyksien toimivuutta lähde- ja kohdejärjestelmien välillä. Alustavassa haussa tietovarastoon kirjoitetaan sisään historiallista dataa, kuten historialliset myyntitapahtumat. Historiallisella voidaan tarkoittaa esimerkiksi aikajaksoa tästä päivästä kaksi vuotta taaksepäin. Alustavan haun jälkeen tietovarastoon kirjoitetaan dataa tasaisin väliajoin, vaikkapa päivittäin. Tällä tavalla tietovarastossa oleva data säilyy ajankohtaisena ja esimerkiksi päivittäisiä myyntejä voidaan seurata ja analysoida.

On useita erilaisia menetelmiä toteuttaa tiedon hakuvaihe. Se voidaan toteuttaa esimerkiksi käyttämällä Amazonin Simple Queue Serviceä (SQS) [3]. SQS:ssä dataa voidaan tallentaa pilvessä oleviin FIFO-jonoihin (first in first out). Haku tietovarastoon toimii tällöin jakamalla lähde- ja kohdejärjestelmien ylläpitäjille avaimet, joilla molemmat pääsevät käsiksi jonoihin. Kun dataa halutaan siirtää lähdejärjestelmästä kohdejärjestelmään, lähdejärjestelmä lisää jonoon dataa ja kohdejärjestelmä hakee jonosta dataa. Yksi vaihtoehto haku-vaiheen toteutukseen on FTP-palvelin, johon sekä lähdeettä kohdejärjestelmien ylläpitäjillä on pääsy. Lähdejärjestelmästä voidaan lähettää dataa sovitussa standardissa, esimerkiksi CSV-standardissa [36] FTP-palvelimelle, josta kohdejärjestelmä hakee dataa tietyin aikavälein.

Datan hakuvaihe on tietovarastohankkeiden suurin kustannustekijä. Datan hakuvaiheen toteuttaminen vie yli 80% tietovarastohankkeiden ajasta, sekä tuo hankkeisiin yli 50% ennakoimattomista kuluista [44]. Ennakoimattomia kuluja on erilaisia, kuten heikko datan laatu lähdejärjestelmässä, datan omistamisoikeuksiin liittyvät säännökset tai vanhentunut teknologia.

Muunnosvaiheessa (engl. transform) haettua dataa muokataan, jotta se saadaan tietovaraston vaatimaan muotoon [10]. Tietoa voidaan tallentaa väliaikaisiin tiedostoihin tai tietokantaan [10, 30]. Muunnosohjelmien (engl. transformation function) avulla voidaan päättää, mitä sarakkeita kirjoitetaan sisään tietovarastoon kirjoitusvaiheessa ja mitkä sarakkeista ohitetaan. Esimerkiksi jos tuotetietoja saadaan useammasta eri lähteestä, muunnosohjelmat etsivät kaikki tuotetiedot ja jäsentävät kaiken tuotedatan yhteen. Joskus tietovarastoa käyttävän tahon on hankalaa muodostaa erillistä tuotetai myymälä -dataa. Kaikki data lähetetään kerralla esimerkiksi myyntitapahtumien mukana. Tämä voidaan välttää datan osioimisella muunnosvaiheessa. Eri osien avulla voidaan esimerkiksi päivittää tuote-, myymälä- sekä myyntitauluja.

Muunnosvaiheessa aineistoa voidaan puhdistaa tai muokata automaattisesti [10]. Dataa voidaan jäsennellellä, jotta siitä saataisiin johdonmukaista ja ehjää [10]. Lisäksi dataa voidaan muokata asettamalla siihen liiketoiminnallisia sääntöjä, kuten *myynti euroissa = myynti kappaleissa * yksikköhinta euroissa*.

Lähdejärjästelmästä voi saapua esimerkiksi seuraavanlainen CSV-standardissa oleva tiedosto:

```
tuotekoodi,tuotenimi,myynnin_aloitus_pvm
1234,suklaalevy,31.12.2017
```

Tietovarastossa voidaan käyttää päivämääräformaattia vvvv-kk-pp, jolloin muunnosohjelmat käsittelevät toisella rivillä olevan datan ”31.12.2017” muotoon ”2017-12-31”. Muunnosohjelman jälkeen tiedosto on valmiina kirjoitettavaksi tietovarastoon:

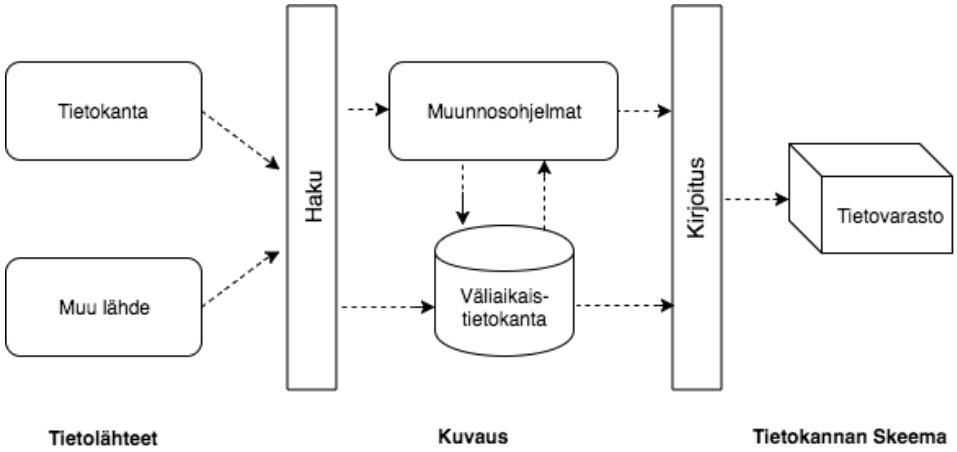
```
tuotekoodi,tuotenimi,myynnin_aloitus_pvm
1234,suklaalevy,2017-12-31
```

Kirjoitusvaiheen (engl. load) alussa data on puhdistettu ja jäsennelty. Kirjoitusvaiheen toteutus vaihtelee, mutta data voidaan kirjoittaa tietovarastoon esimerkiksi suoraan muunnosohjelmien jälkeen tai väliaikaisen tietokannan avulla [10]. Tässä vaiheessa ETL-prosessia data on osioitettu eli data on hajoitettu esimerkiksi erillisiin tuote-, myymälä- ja myyntitapahtumatietoihin. Nämä tiedot ovat tietovaraston haluamassa muodossa. Kirjoitusvaiheessa data kirjoitetaan tietovaraston tietokantatauluihin. Onnistuneen kirjoitusvaiheen jälkeen dataa voidaan analysoida tietovaraston käyttöliittymästä.

Kuvassa 3 on ETL-prosessin malliehdotus [10]. Ehdotus tuo esille ETL-prosessin vaiheet tarkemmalla tasolla kuin kuvassa 2. Mallissa ETL-prosessi on jaoteltu kolmeen osavaiheeseen, jotka ovat datalähteet, kuvaus (engl. mapping), sekä tietovaraston skeema. Lähdevaiheessa data on voi olla useassa eri lähdejärjestelmän tietokannassa. Muunnosvaiheessa dataa prosesoidaan ja sitä voidaan säilöä väliaikaisessa tietokannassa. Tietovaraston skeemavaiheessa lähdejärjestelmästä saatu data on prosessoitu ja päätyy lopulta tietovarastoon.

3 Datan laatu

Korkealaatuiseen dataan pyrkiminen on kalleinta ja aikaavievintä tietovarastohankkeissa [16]. Täydelliseen laatuun ei kannata pyrkiä, eikä se ole tarpeellista, mutta on tärkeää pystyä esittämään tietovarastoa käyttävän yrityksen data sillä tasolla, jolla yritys saa mahdollisimman paljon lisäarvoa tietovaraston käytöstä. Tietovaraston käyttäjien ja tietovarastohankkeen johtajien on mietittävä, mitä hankkeelta vaaditaan ja mitä sillä yritetään



Kuva 3: Tarkempi malli ETL-prosessista [10].

saavuttaa. Tietovarastoa käyttävän yrityksen on tutkittava, mitkä osa-alueet datasta ovat elinehto yrityksen liiketoiminnan kasvun kannalta [5].

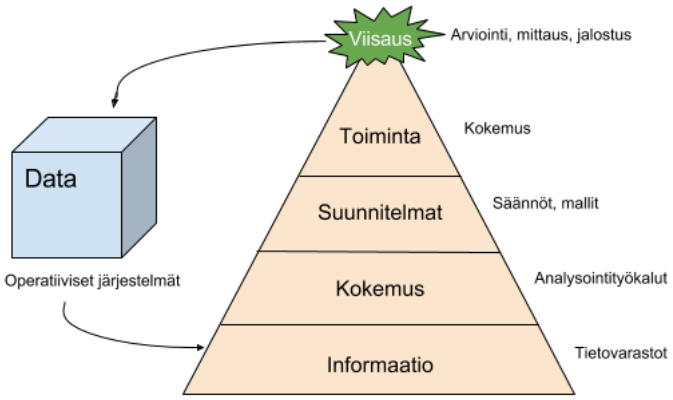
Tämä luku keskittyy määrittämään, mitä hyvä- ja huonolaatuinen data tarkoittaa eri konteksteissa. Ensin luvussa 3.1 käydään läpi, miksi hyvälaatuiseen dataan on hyvä pyrkiä liiketoiminnallisesta näkökulmasta. Tämän jälkeen luvussa 3.2 määritellään datan laatu käyttäjän näkökulmasta ja luvussa 3.3 tuodaan esille tietotekninen näkökulma.

3.1 Huonolaatuisen datan liiketoiminnalliset seuraukset

Huonolaatuinen data vaikuttaa negatiivisesti eri yritysten tai järjestöjen toimintaan [31, 9]. Jos asiakkaiden tiedot, kuten puhelinnumero, osoite tai tilinumero, ovat väärin, yritysten on lähes mahdoton tehdä kauppaa. Yhdysvaltalainen Data Warehous Institute arvioi yritysten menettävän vuodessa yli 600 miljardia dollaria virheellisen datan vuoksi [9]. On arvioitu, että 1-5% datasta on virheellistä ellei datan laatuun olla nähty merkittävästi vaivaa [31].

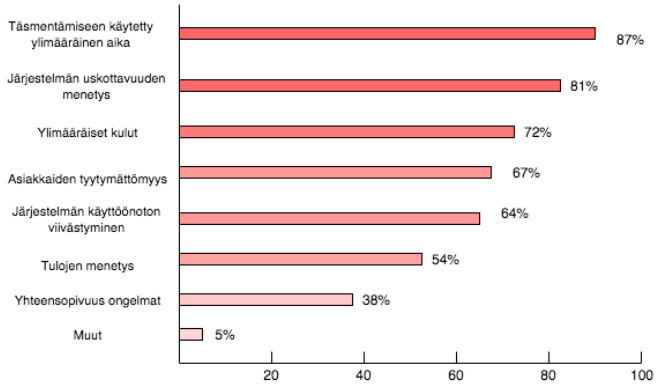
Kuvassa 4 esitellään lähdejärjestelmän ja tietovaraston välinen datakierto [9]. Lähdejärjestelmästä tuleva data tallennetaan tietovarastoon. Analyytikot arvioivat tietovarastossa olevaa dataa, kehittävät suunnitelmia ja jakavat ideoita yrityksen johdolle. Yrityksen johto toimeenpanee suunnitelmat, joista saadaan kokemuksia ja viisautta. Nämä suunnitelmat muokkaavat yrityksessä muodostuvaa dataa, kuten ostopäätöksiä, joka päätyy tietovarastoon seuraavissa hakuoperaatioissa. Jos tietovarastoon päätyy virheellistä dataa, virheet kumuloituvat ja vääriä päätöksiä tehdään huonolaatuisen datan vuoksi.

Seuraava esimerkki on artikkelista [9]. Vakuutusyhtiö saa 2 miljoonaa korvausvaatimusta kuukaudessa. Yksi vaatimus pitää sisällään 377 datakenttää. Oletetaan että 0.001% datakentistä on virheellisiä. Tällöin virheellisiä data-



Kuva 4: Datan kiertokulku. Data saapuu lähdejärjestelmästä tietovarastoon analysoitavaksi. Tietovarastossa olevan datan ja analyysien perusteella tehdään päätöksiä, jotka vaikuttavat liiketoimintaan, sekä liiketoimintaa kuvaavaan dataan [9].

kenttiä on kuukaudessa 754000 ja vuodessa yli 9 miljoonaa. Vakuutusyhtiö arvioi kaikista kentistä 10% olevan liiketoiminnan kannalta kriittisiä, jolloin vuositasolla korjattavia kenttiä on 900000. Jos vakuutusyhtiö arvioi yhden virheen maksavan 10 euroa kappaleelta, vuositason tappiot datavirheiden takia ovat 9 miljoonaa euroa.



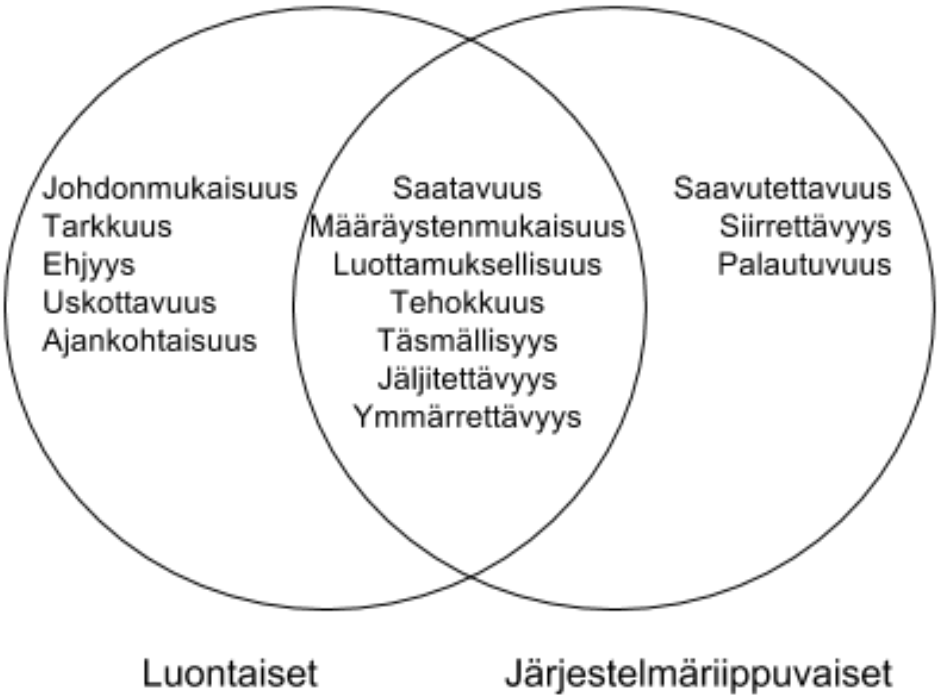
Kuva 5: Huonolaatuisen datan vaikutuksia yritykselle. Vastaukset pohjautuvat 286 osallistujaan, jotka saivat valita useamman vastauksen [9].

Kuvassa 5 eritelty huonosta datan laadusta johtuvia ongelmia. Ongelmat pohjautuvat kyselyyn, johon osallistui 286 vastaajaa [9]. Vastauksien perusteella kaksi yleisintä ongelmaa ovat datan täsmentämiseen käytettävä ylimääräinen aika (87%), sekä järjestelmään uskottavuuden menetys (81%). Suurin osa vastaajista myös ilmaisi yrityksen menettävän tuloja (54%). 67% vastaajista ilmoitti asiakkaiden tulevan tyytymättömiksi huonolaatuisen da-

tan takia.

3.2 Datan laatu käyttäjän näkökulmasta

Huonolaatuisella datalla voi olla merkittäviä ekonomisia vaikutuksia [31, 43], mutta milloin data on hyvä tai huonolaatuista? ISO/IEC 25012 datan laatumallissa esitellään 15 datan laatupiirrettä, jotka on jaettu luontaisiin ja järjestelmäriippuvaisiin datan laatupiirteisiin [20]. Laatupiirteet ovat esitelty kuvassa 6. Luontaiset datan laatupiirteet ovat kontekstisidonnaisia.



Kuva 6: ISO/IEC 25012 datan laatumallin laatupiirteet [20]. Kuvassa vasemmassa ympyrässä ovat luontaiset laatupiirteet (engl. inherent data quality characteristics) ja oikeassa järjestelmäriippuvaiset laatupiirteet (engl. system dependent data quality characteristics). Ympyröiden leikkausalueella ovat laatupiirteet, jotka ovat sekä luontaisia -, että järjestelmäriippuvaisia laatupiirteitä.

Ne liittyvät johonkin aihealueeseen, kuten liiketoimintaan ja niissä voi olla liiketoimintaan liittyviä sääntöjä tai rajoitteita. ISO 25012 standardin luontaiset laatupiirteet ovat *tarkkuus* (engl. accuracy), *ehjyys* (engl. completeness), *johdonmukaisuus* (engl. consistency), *uskottavuus* (engl. credibility) ja *ajankohtaisuus* (engl. currentness) [20].

Johdonmukaisuudella tarkoitetaan datan olevan yhtenäistä joko itsensä tai muuhun verrattavissa olevan datan kanssa [20]. Esimerkiksi tuotteiden

väri on esitetty sanana (”musta”) tai heksakoodina (”#000000”).

Datan *tarkkuudella* tarkoitetaan, kuinka lähellä datan arvo on todellista arvoa [20]. Jos tuotteen todellinen yksikköhinta on 1.35 euroa lähdejärjestelmässä, mutta tietovarastoon yksikköhinnaksi on saatu 1.4 euroa, onko tuotteen hinta esitetty tarkasti? Jos tuotetta myydään vuoden aikana miljoona kappaletta ja myyntien arvo lasketaan *myydyt kappaleet * yksikköhinta*, myyntien todellinen arvo on vuodessa 1 350 000 euroa, kun tietovarastoon laskettu arvo on 1 400 000 euroa.

Tarkkuus jaetaan syntaksiseen tarkkuuteen (engl. syntactic accuracy) ja semanttiseen tarkkuuteen (engl. semantic accuracy) [20]. Syntaksinen tarkkuus tarkoittaa datan sopivan aihealueeseen, mutta se ei ole oikein. Esimerkiksi tietovarastoon vastaanotettu myyntihinta on liukuluku, mutta hinta ei vastaa todellista hintaa. Semanttinen tarkkuus tarkoittaa datan olevan merkitykseltään oikein, mutta sen syntaksi on väärin. Esimerkiksi tuotteen hinta on datassa ”kolme euroa”, mutta tuotteen hinnan datatyyppi on tietovarastossa liukuluku.

Ehjyydellä tarkoitetaan kaikkien datakenttien sisältävän arvon, joiden kuuluu sisältää arvo kyseisessä kontekstissa [20]. Jos data ei ole ehjää, sen käyttäjät eivät voi suoriutua vaadituista tehtävistä. Esimerkiksi tuotannonsuunnittelija haluaa tietää, kuinka paljon tuotetta A on valmistettava tehtaassa Y ensi viikolla. Tuotteella A alkaa kampanja ensi viikolla, jolloin tuotetta myydään normaalia enemmän. Tuotannonsuunnittelijan tarvitsee tietää tuotteen keskimääräinen viikoittainen myyntimäärä ja arvion, kuinka paljon ensi viikolla alkava kamppanja lisää tuotteen myyntiä. Arvion voi muodostaa esimerkiksi aiemman vastaavanlaisen kampanjan vaikutuksesta myyntimäärään. Kampanjalisäys ja viikoittainen keskimääräinen myyntimäärä kertoo, kuinka paljon tuotetta on valmistettava.

Ajankohtaisuudella tarkoitetaan datan olevan sopivan ikäistä, että se säilyttää käyttökelpoisuutensa kyseisessä kontekstissa [20]. Esimerkiksi sanomalahden toimittajan tulee tietää asiakkaiden nykyiset osoitteet entisten sijaan.

Laatupiirteet, jotka ovat riippuvaisia käyttötarkoituksesta ja järjestelmästä, ovat saatavuus (engl. accessibility), määräystenmukaisuus (engl. compliance), luottamuksellisuus (engl. confidentiality), tehokkuus (engl. efficiency), täsmällisyys (engl. precision), jäljitettävyys (engl. traceability), ymmärrettävyys (engl. understandability) [20].

Saatavuudella tarkoitetaan dataan olevan pääsy käyttötarkoitusta varten varsinkin käyttäjille, joilla on jokin rajoite [20]. Esimerkiksi jos data esitetään värillisinä graafeina, on hyvä huomioida erilaiset värisokeudet.

Määräystenmukaisuudella tarkoitetaan datan seuraavan sovittuja standardeja ja konventioita. Jos tietovarastoon vastaanotettu data ei seuraa sovittuja standardeja, sitä ei mahdollisesti voida kirjoittaa tietovarastoon, jolloin dataa ei voi käyttää.

Luottamuksellisuudella tarkoitetaan dataan olevan pääsy vain asianomai-

silla käyttäjillä [20]. Esimerkiksi pankkijärjestelmien käyttäjillä ei ole pääsy muihin kuin omiin tilitapahtumiin.

Tehokkuudella viitataan datan prosessoinnin suoritusnopeuteen [20]. Esimerkiksi näkymien latausajat kertovat datan hakemisen tehokkuudesta. Data pitää prosessoida tehokkaasti, jotta käyttäjien ei tarvitse odottaa varsinkin käyttäessä järjestelmiä, joissa aika on kriittinen tekijä. Esimerkiksi tilausjärjestelmissä aika voi olla kriittinen tekijä.

Datan on hyvä olla *ymmärrettävää* ja yksiselitteistä, jotta sen käyttäjät eivät joudu tilanteeseen, missä dataa voidaan tulkita monin eri tavoin. Data esitetään käyttötarkoitukseen sopivalla kielellä, symboleilla ja terminologialla [20]. Esimerkiksi käyttäjälle näytetään tuotteiden hinnat hänen oman maansa valuuttayksikössä.

Järjestelmäriippuvaiset laatupiirteet liittyvät tietokonejärjestelmään, missä dataa analysoidaan, käsitellään tai säilötään. Järjestelmäriippuvaiset laatupiirteet ovat sidoksissa teknologiaan, jolla tietokonejärjestelmä on tehty. Järjestelmäriippuvaiset laatupiirteet ovat saavutettavuus (engl. availability), siirrettävyys (engl. portability) ja palautuvuus (engl. recoverability) [20].

On useita eri syitä, milloin data voi olla heikosti *saavutettavissa* [38]. Jos datan hakemiseen käytetään liian vähän resursseja, voidaan päätyä tilanteeseen, jossa datasta saadaan vain jokin osajoukko. Resursseilla tarkoitetaan esimerkiksi internet-yhteyttä tai tietokoneen prosessointitehoa. Datalle voidaan asettaa käyttörajoituksia turvallisuussyistä, jolloin dataa ei välttämättä saada haettua ollenkaan tai sitä ei saada riittävästi. Datan saatavuutta heikentävät tilanteet, joissa dataa on paljon ja sitä halutaan siirtää järjestelmien välillä. Data ei välttämättä ehdi siirtyä järjestelmästä toiseen riittävän nopeasti, että siitä olisi datan käyttäjälle hyötyä.

Tietovarastoissa dataa voidaan käsitellä käyttöliittymän avulla, jolloin tietovarastoa käyttävän henkilön on mahdollista muokata dataa virheellisellä tavalla. Tietovarasto-ohjelmistoissa ylläpidetään muutoshistoriaa ja varmuuskopioita, joiden avulla data on *jäljitettävää* [2]. Muutoshistoria ilmaisee, onko tietokannan rivi muuttunut käyttäjän vai järjestelmän toimesta. Käyttäjille voidaan antaa rajoitetut käyttöoikeudet, joiden avulla datan muokkausoikeuksia ja saavutettavuutta voidaan rajoittaa. Virheen sattuessa varmuuskopioiden avulla voidaan palauttaa järjestelmän data aiempaan tilaan (*palautuvuus*).

Siirrettävyydellä tarkoitetaan datan säilyttävän laatunsa, kun se asennetaan, korvataan tai siirretään järjestelmästä toiseen [20]. Esimerkiksi tuotannonsuunnittelija haluaa lähettää myyntiennusteita tehtaalle. Myyntiennusteiden avulla tehtaan työntekijät tietävät, kuinka paljon tuotteita on valmistettava. Tehtaan työntekijöiden on vastaanotettava myyntiennusteet samassa täsmällisessä ja tarkassa muodossa kuin ne ovat tietovarastossa pystyäkseen valmistamaan tuotteita tarpeen mukaan.

3.3 Datan laatu tietoteknisestä näkökulmasta

Datan laatuongelmat jaotellaan yhden ja useamman datalähteen laatuongelmiksi [6, 26, 30]. Useat tutkimukset jaottelevat yhden lähteen ja useamman datalähteen virheet skeema-, ja instanssitason laatuongelmiin [6, 30]. Skeematason laatuongelmat liittyvät tietokannan skeemaan ja instanssitason laatuongelmat datan sisältöön.

3.3.1 Skeematason datan laatu

Tietokannan skeemalla tarkoitetaan tietokannan määrittelykokonaisuutta, joka pitää sisällään tietokantataulujen väliset suhteet eli relaatiot. Tietokantaa suunniteltaessa luodaan joukko eheysrajoitteita (engl. integrity constraints), joiden avulla pystytään karsimaan huonolaatuista dataa [6]. Näitä eheysrajoitteita ovat arvojoukkoeheys (engl. domain integrity), avaimen täydellisyys (engl. entity integrity) ja viite-eheys (engl. referential integrity) [45].

Arvojoukkoeheydellä tarkoitetaan tietokantasarakkeen kuuluvan johonkin arvojoukkoon. Esimerkiksi sarakkeelle *ikä* annetaan arvojoukoksi kokonaislukujoukko [0,120]. Avaimen täydellisyyssäännöllä tarkoitetaan, että pääavaimet (engl. primary key) eivät saa olla tyhjiä ja niiden pitää sisältää sopivia arvoja. Jos pääavain on tyhjä, tietokantojen rivejä ei voida tunnistaa toisistaan. Muissa kuin pääavaimissa voidaan sallia tyhjiä arvoja.

Viite-eheydellä tarkoitetaan tietokantataulujen välistä yhteyttä. Esimerkiksi *Tuote*-taulussa voi olla viite *Tuoteryhmä*-tauluun. Viite muodostetaan lisäämällä *Tuote*-tauluun viitesarake. Viitesarakkeen arvo on jokin *Tuoteryhmä*-taulun pääavain. Pääavaimen on oltava oikein, tai muuten yhteyttä ei voida muodostaa.

Seuraavat datan laatuongelmat voidaan välttää hyvän tietokantasuunnittelun seurauksena [6]:

- Puutteellisen datan voi välttää asettamalla sarakkeelle ehdon, ettei puuttuvia arvoja hyväksytä (arvojoukkoeheys). Esimerkiksi työntekijän sosiaaliturvatunnus ei voi olla tyhjä.
- Väärät datatyypit estetään asettamalla sarakkeelle datatyyppi (arvojoukkoeheys). Esimerkiksi työntekijän ikä ei saa sisältää merkkejä.
- Mahdottomia arvoja voidaan vähentää antamalla sarakkeelle arvoväli, mihin arvon on kuuluttava (arvojoukkoeheys). Esimerkiksi työntekijän iän on oltava välillä [18, 65].
- Toisistaan riippuva data hylätään, jos viitteet ovat väärin (viite-eheys). Esimerkiksi työntekijän osaston nimi tai tunnus ei voi olla mikään arvo, joka ei kuulu *Osasto*-tauluun.
- Avainten duplikaattiarvoja ei hyväksytä (avaimen täydellisyys). Taulu-

jen pääavaimet ovat uniikkeja arvoja. Esimerkiksi kahdella eri työntekijällä ei voi olla samaa sosiaaliturvatunnusta.

3.3.2 Instanssitason datan laatu

Instanssitason datan laatuvirheet tarkoittavat virheitä, joita ei pystytä havainnoimaan skeematasolla. Instanssitason virheet liittyvät datan sisältöön ja merkitykseen. Joissain tapauksissa instanssitason ongelmat voivat johtua huonosta tietokannan skeeman suunnittelusta. Esimerkiksi tietokantaan voi päätyä tyhjiä arvoja, missä niitä ei saisi olla. Instanssitason ongelmat jaetaan yhden tietueen (engl. single record), sekä usean tietueen (engl. multiple record) ongelmiin [6, 26, 30].

Yhden tietueen ongelmat tarkoittavat virheellistä dataa, joka ei ole riippuvaista mistään muusta datasta tietokannassa [6]. Yleinen ongelma on esimerkiksi malliksi lisätty data. Esimerkiksi henkilötunnuskenttään on lisätty arvo 99999, joka ei ole kenenkään henkilön oikea henkilötunnus. Toisaalta ihmiset voivat syöttää vahingossa vääriä arvoja. Esimerkiksi iäksi voidaan vahingossa syöttää 26, vaikka henkilön oikea ikä on 27. Yksi mahdollinen virhetilanne on virkanimikkeiden sulauttaminen nimikenttään (”Presidentti Tarja Halonen”). Käyttäjät voivat myös vahingossa syöttää tietoja vääriin kenttiin. Esimerkiksi ”Kaupunki” –kenttään voi vahingossa syöttää arvon ”Suomi”. Joissain tapauksessa on hyvä varmistaa, että käyttäjät syöttävät tiedot tietokannalle sopivassa formaatissa. Esimerkiksi päivämäärille on olemassa useita eri formaatteja. Kaikki tulkinnanvaraisuus on hyvä yrittää karsia. Esimerkiksi käyttäjä voi kirjoittaa nimen muodossa ”M. Tahvanainen”, jossa M voi tarkoittaa ”Matti” tai ”Markku”.

Useamman tietueen ongelmat tarkoittavat toisiinsa liittyvien tietokantarivien virheellisyyttä yhdessä tai useammassa tietokantataulussa. Esimerkiksi *Henkilö*–taulussa voi olla kaksi samaa henkilöä kuvaavaa riviä, koska syntymäajat on syötetty kahdella eri tavalla. Ensimmäinen henkilö on "Matti Tahvanainen, syntynyt 7.11.1990", kun taas toinen "Matti Tahvanainen, syntynyt 07/11/1990". Käyttäjälle on tärkeää ilmaista, missä muodossa tiedot halutaan, jolloin formaattiongelmat vältetään. Toinen esimerkki on rahaa koskevissa kentissä. Rahasymbolin paikka voidaan ilmaista ennen tai jälkeen arvon, kuten $4.00 tai 4.00$. Desimaalierottimena voidaan käyttää pilkkua tai pistettä. Mittayksikkönä voidaan käyttää tuumia tai senttimetrejä. Tietokannan haluama muoto on hyvä lyödä lukkoon, jonka avulla voidaan estää virheelliset syötteet.

3.3.3 Yhden lähteen laatuvirheet

Laatuvirheitä ja niistä johtuvia ongelmia voi ilmetä tietovarastohankkeen monissa eri vaiheissa, kuten ETL-prosessissa, lähdejärjestelmässä tai tietovarastoon kirjoitetun datan analysoinnissa. Useat tutkimukset jaottelevat

laatuongelmat yhden sekä useamman datalähteen ongelmiksi [6, 26, 30]. Taulukossa 1 tuodaan esille esimerkkejä yhden lähteen datavirheistä. Seuraavaksi selvennetään kohtia taulukosta 1.

Taulukon 1 rivillä kaksi oleva kuvaus *Virheellinen syntaksi* voi tarkoittaa esimerkiksi päivämäärän virheellistä muotoilua [26]. Rivillä kuusi oleva *Sopimaton konteksti* tarkoittaa kentän tai kolumnin otsakkeeseen suhteutettuna sopimatonta arvoa [6, 26]. Esimerkiksi *Postinumero*-kenttään on syötetty arvo "Helsinginkatu". Vastaavasti kohdalla 8 voidaan tarkoittaa esimerkiksi "asdasdasd"-arvon syöttämistä kenttään *Nimi* [26]. Rivillä 11 oleva synonyymit tarkoittaa tilannetta, jossa sarakkeessa saatu arvo on kirjoitusasultaan eroavainen, mutta semantiikaltaan sama kuin jo olemassa oleva arvo [6, 26, 30]. Esimerkiksi kolumni *Ammatti* sisältää arvon "Lehtori" ja "Opettaja", millä pyritään ilmaisemaan samaa ammattia eri sanoin.

Tietokannan taulun sarakkeille voidaan määritellä sääntöjä. Esimerkiksi *Henkilön nimi* sarakkeen tulee sisältää ainakin kaksi sanaa [26]. Tätä kutsutaan määrittelyjoukoksi, mihin rivillä 13 oleva *Määrittelyjoukon ulkopuolella oleva arvo* viittaa.

Taulukon 1 rivi 14 viittaa tapauksiin, missä rivillä olevien sarakkeiden välinen yhteys on epäjohdonmukainen [6, 30]. Esimerkiksi myyntien arvo euroissa on laskettu myyntihinnan sekä myytyjen kappaleiden avulla: *myyntien arvo euroissa = myyntihinta euroissa * myydyt kappaleet*, mutta erikseen vastaanotettu myyntihinta ja myydyt määrät eivät vastaa myyntien arvoa.

Tietovaraston käyttävän yhtiön on vaalittava oman liiketoimintansa kuvaamiseen ja hallinnoimiseen tarvittavaa dataa, mihin taulukon 1 rivi 15 viittaa [6]. Jos yhtiö ei ole huolellisesti tallentanut ja säännöllisesti tarkistanut yhtiön liiketoiminnan kannalta tärkeää dataa, tiedon tallentamisesta tietovarastoon ei ole mitään hyötyä yhtiölle.

Jos datalähteestä tulevassa tiedostossa on väärät merkistöasetukset (kohta 16) [30], sen lukeminen tietovarastoon voi aiheuttaa ongelmia. Esimerkiksi ääkköset ovat yksi esimerkki ongelmallisista kirjaimista, jos niihin ei osata varautua. Tietovaraston käyttäjän ja palveluntarjoajan välille voidaan tehdä sopimus, mitä merkistöasetusta käytetään, jolloin vältytään merkistöasetuksiin liittyviltä ongelmilta. Jos käyttäjä haluaa ääkköset näkyviin, merkistöasetuksena voidaan käyttää esimerkiksi UTF-8 standardia.

Erikoismerkeillä on yleensä jokin käyttötarkoitus tietovarastoissa (kohta 17) [6]. Esimerkiksi pilkkua voidaan käyttää arvojen erottamiseen CSV-standardissa [36]. Jos pilkulla on arvojen erottava toiminto, sitä ei voi käyttää arvoalueen osana. Jos pilkku halutaan datakentän sisään, se käsitellään lainausmerkin avulla.

Tietovarastoon halutaan kirjoittaa seuraavanlainen yhtä tuotetta kuvaava CSV-standardissa oleva tiedosto, jossa arvoalueet ovat pilkuin eroteltuna:

```
koodi,nimi,hinta
1234,Suo, kuokka ja Jussi, 21.90
```

Taulukko 1: Yhden datalähteen laatuvirheet. Taulukossa *Taso*-sarakkeessa oleva arvo *S* tarkoittaa skeematason laatuvirhettä ja *I* tarkoittaa instanssitason laatuvirhettä. Jos sarakkeessa on molemmat arvot *S* ja *I*, laatuvirhe voi olla sekä skeema- että instanssitasoinen. Taulukon laatuvirheitä on kerätty tutkimuksista [6, 26, 30].

	Virheen kuvaus	**Taso**	**Esimerkki**
1	Puuttuva arvo	S, I	Tyhjä tuotekoodi
2	Virheellinen syntaksi	S, I	Päivämäärä annettu muodossa kk/pp/vvvv. Haluttu muoto on vvvv-kk-pp.
3	Vanhentunut arvo	I	Vanhentunut osoite
4	Väärä arvoväli	S	Henkilön ikä on -3
5	Kirjoitusvirheet	I	Nimi: Mtti
6	Sopimaton konteksti	I	Osoite: Matti Tahvanainen
7	Useita arvoja yhdessä kentässä	S, I	Nimi: Matti 27 vuotta
8	Merkityksetön arvo	I	N\A
9	Monikäsitteinen arvo	I	Nimi: M. Tahvanainen
10	Usea uniikki arvo	S	Kaksi tuoteriviä samalla tuotekoodilla
11	Synonyymit	I	Kaksi samaa tarkoittavaa tuoteryhmää 'Oluet' ja 'Kaljat'
13	Määrittelyjoukon ulkopuolella oleva arvo	S, I	Nimeen syötetty 5 sanaa, mutta halutaan 2.
14	Epäjohdonmukainen arvo	I	Tuote *banaani* kuuluu tuoteryhmään *juomat*
15	Validointirutiinien puuttuminen lähdejärjestelmässä		
16	Väärät merkistöasetukset	S	Haluttu merkistöasetus on UTF-8, mutta vastaanotettu asetuksella ISO-8859-1
17	Epäjohdonmukainen erikoismerkkien käyttö	S, I	1$ vs $1

Taulukko 2: Usean datalähteen laatuvirheitä [6, 26, 30].

	Virheen kuvaus	**Taso**
1	Syntaksien epäjohdonmukaisuus	S, I
2	Eroavat mittayksiköt	S, I
3	Kuvaamisen epäjohdonmukaisuus	I
4	Eroavat aggregaatiotasot	S, I
5	Homonyymit arvot	I
6	Määrittelyjoukon ulkopuolella oleva arvo	S, I
7	Duplikaatit arvot	S, I

Tässä tapauksessa nimi-kenttään halutaan kirjoittaa koko merkkijono "Suo, kuokka ja Jussi". Tuotteen nimi sisältää pilkun, jolloin datarivillä on neljä arvoa halutun kolmen sijaan: "1234", "Suo", "kuokka ja Jussi"ja "21.90". Nimikenttä on käsiteltävä lainausmerkeillä, jolloin tiedosto saadaan seuraavaan muotoon:

```
koodi,nimi,hinta
1234,"Suo, kuokka ja Jussi", 21.90
```

Tällöin tuotekoodille 1234 saadaan oikea nimi ja hinta.

3.3.4 Usean datalähteen laatuvirheet

Tietovarastoa käyttävällä yhtiöllä voi olla käytössä useita eri tietokantoja. Syy tähän voi olla esimerkiksi kansainvälinen yhtiö, joka on hajauttanut järjestelmänsä jokaiselle maalle, missä yhtiö harjoittaa liiketoimintaa. Usean datalähteen laatuvirheet tarkoittavat ominaisia datan laatuvirheitä käytettäessä useampaa kuin yhtä datalähdettä. Taulukkoon 2 on kerätty usean datalähteen laatuvirheitä monista eri tutkimuksista [6, 26, 30]. Datalähdeongelmia, mitkä ovat riippumattomia datalähteiden määrästä, ovat muun muassa synonyymit, ylimääräiset arvot, sekä epäjohdonmukaiset arvot.

Syntaksien epäjohdonmukaisuudella (taulukon 2 rivi 1) tarkoitetaan tilannetta, jossa sarakkeessa oleva arvo on kirjoitettu eri syntaksilla haettaessa arvoja useasta eri datalähteestä [26]. Esimerkiksi ensimmäisestä datalähteestä saatu päivämäärä on muodossa *yyyy-mm-dd* ja toisesta lähteestä se saadaan muodossa *yyyy/mm/dd*. Eroavat mittayksiköt (rivi 2) tarkoittaa esimerkiksi tilannetta, jossa ensimmäisestä datalähteestä saadut arvot ovat euroissa ja toisesta datalähteestä saadut arvot ovat dollareissa [26, 30]. Kohdalla kolme tarkoitetaan tilannetta, jossa esimerkiksi sukupuoli kuvataan ensimmäisessä lähteessä merkein ”M ” ja ”N”ja toisessa lähteessä ”Mies” ja ”Nainen” [30].

Aggregaatiotasoilla (taulukon 2 rivi 4) tarkoitetaan, millä tarkkuusasteella data esitetään [26, 30]. Esimerkiksi myyntimäärä *Tuote*-tasolla tarkoittaa

tuotteiden myyntiä, kun taas myyntimäärä *Tuote-toimipiste*-tasolla tarkoittaa tuotteiden myyntiä toimipisteissä. *Tuote-toimipiste*-taso ilmaisee myynnit tarkemmin kuin *Tuote*-taso. Lähdejärjestelmien on hyvä ilmaista samaa dataa samoilta aggregaatiotasoilta.

Rivillä viisi olevat *Homonyymit arvot* tarkoittavat arvoja, jotka ovat kieliasultaan samat, mutta semantiikaltaan eroavat [26, 30]. Esimerkiksi kuusi (puulaji) ja kuusi (luku).

Kohta kuusi esiteltiin jo aiemmin yhden datalähteen ongelmana, mutta se on myös useamman datalähteen ongelma pienellä erolla [26]. Yritys voi esimerkiksi asettaa liiketoiminnalliseksi säännöksi, että tuoteryhmiä on yhteensä 20 kappaletta. Ensimmäisessä datalähteessä tuoteryhmiä on 12 kappaletta ja toisessa 15 kappaletta. Kun uniikit tuoteryhmät yhdistetään tietovarastoon molemmista lähteistä, huomataan niitä olevan yhteensä 23 kappaletta.

Rivillä 7 tarkoitetaan duplikaattiarvoja kahdessa eri lähteessä [6, 30]. Duplikaattidatan havaitseminen on ollut pitkään tiedemaailmaa askarruttava aihe, ja sitä on tutkittu paljon [8, 11, 47]. Tässä tutkielmassa duplikaatilla tarkoitetaan kahden tai useamman tietokantarivin semanttista yhdenvertaisuutta eli rivit kuvaavat samaa oikean elämän tapahtumaa [11, 24]. Duplikaattirivit aiheuttavat ongelmia, kun järjestelmä tulkitsee rivit kahdeksi eri riviksi, vaikka ne ovat semanttisesti sama asia, kuten esimerkiksi kaksi samaa myyntitapahtumaa ilmaisevaa riviä. Duplikaatit voidaan jaotella kahteen osaan: strukturaalisiin eli rakenteellisiin virheisiin ja leksikaalisiin eli kieliopillisiin virheisiin [11]. Strukturaalisia virheitä ilmenee tyypillisesti integroitaessa kahta eri tietokantaa. Ensimmäisessä tietokannassa *osoite*-kenttä voi olla yhtenä osana, kun taas toisessa se on jaoteltu *katuosoite*-, *postinumero*- ja *kaupunki*-kenttiin. Leksikaaliset virheet voivat olla esimerkiksi kirjoitusvirheitä. Yksi keino tunnistaa duplikaatteja on muodostaa merkkijonovertailu kahden tai useamman rivin sarakkeiden välillä [11].

4 Datan validointi

Tässä luvussa kerrotaan, mitä datan validoinnilla tarkoitetaan ja kuinka erilaiset yritykset validoivat dataa. Kirjallisuudessa on tuotu esille erilaisia datan validointikeinoja, joista kerrotaan luvussa 4.1. Datan validointiin on ehdotettu ja luotu erilaisia automaatioita ja työkaluja. Näiden ominaisuuksia käydään läpi luvussa 4.2.

4.1 Validointiprosessit

Datan validoinnilla tarkoitetaan datan oikeallisuuden varmistamista. Validointi voidaan kohdistaa esimerkiksi tietokantoihin tai tiedostoihin ja sitä voidaan tehdä esimerkiksi vertailemalla tai etsimällä kirjoitusvirheitä. Datavirheitä tulee väistämättä muun muassa kirjoitusvirheiden muodossa.

Kun useita datalähteitä integroidaan tietovarastoon, dataa voidaan puhdistaa (engl. data cleaning) erilaisten automaatioiden avulla. Puhdistamiseen on ehdotettu muun muassa koneoppimista (engl. machine learning) hyödyntävää menetelmää [14] sekä vaihe vaiheelta eteneviä prosesseja [18, 25]. Kirjallisuudessa useiden datalähteiden integroinnista samaan tietokantaan puhutaan yhdistys- ja puhdistus -ongelmana (engl. merge/purge problem) [17, 18].

Datan validointia on tutkittu Saksalaisten ja Sveitsiläisten yhtiöiden tietovarastohankkeiden eri vaiheissa [16]. Suurin osa (76%) osallistuneista yhtiöistä tekee laadun tarkistuksia ETL-vaiheessa. Useat yritykset tekevät laadun tarkistuksia loppukäyttäjien palautteen avulla (68%). 60% yhtiöistä analysoi datan laatua tietokantakyselyiden avulla, sekä erilaisten standardien ja mallien avulla (50%). Vain 20% tekee laadun tarkistuksia säännöllisesti. Taulukossa 3 on esitelty tapoja, miten tietovarastoa käyttävät yritykset tarkkailevat datan laatua.

Taulukko 3: Erilaisia tapoja tarkistaa datan validiteettia tietovarastoissa [16].

1	Arvovälit ja datatyypit
2	Duplikaatit avainarvojen avulla
3	Viittaavien arvojen ehjyys
4	Viittaavien arvojen puutteellisuus
5	ETL-prosessin virheilmoitukset
6	Uskottavuus
7	Loppukäyttäjien huomiot

Sarakkeille voidaan asettaa arvoväli (taulukon 3 rivi 1), millä alueella kentän arvon on oltava ollakseen oikein. Luvun 3.3.3 esimerkissä tietovarastoon haluttiin kirjoittaa tuote, jonka hinta oli 21.90. Hintakenttään voidaan asettaa arvoväli- ja tyyppitarkistus esimerkiksi ehdolla "tuotteen hinnan tulee olla nollaa suurempi liukuluku". Tuotteen hinnalle voidaan antaa yläraja, esimerkiksi 500.

Duplikaattien tarkistamista on oleellista tehdä, varsinkin liiketapahtumien osalta (rivi 2). Jos tietovarastoon kirjoitetaan tuplana esimerkiksi myyntitapahtumat, saadaan harhaanjohtava käsitys yrityksen liiketoiminnasta. Duplikaatit voidaan tunnistaa avainarvojen avulla, jos esimerkiksi jokaisella myyntitapahtumalla on oma uniikki id-kenttä. Tällöin voidaan tarkistaa myyntitapahtuman id:llä, onko myyntitapahtuma jo aiemmin kirjoitettu tietovarastoon. Jos myyntitapahtuma on tietovarastossa, myyntitapahtumaa ei kirjoiteta uudelleen tietovarastoon ja virhe ilmoitetaan loppukäyttäjälle. Duplikaatit voi myös tarkistaa rivimäärien avulla vertaamalla tietovarastoon kirjoitettuja rivimääriä lähteeseen.

Viitearvoja voidaan käyttää heijastamaan esimerkiksi tilannetta, jolloin

jokin tuote korvaa toisen tuotteen. Korvattava tuote viittaa korvaavaan tuotteeseen, jolloin voidaan esimerkiksi ennustaa korvaavan tuotteen tulevia myyntejä, jos korvattava ja korvaava tuote ovat liiketoiminnallisesti samantyylisiä eli oletetut myyntimäärät ovat samaa suuruusluokkaa. Viitteet eivät ole silloin ehjiä, kun korvattava tuote viittaa joko väärään tuotteeseen tai tuotteeseen, joka ei ole olemassa (taulukon 3 rivi 3).

Tietokantataulujen välillä käytetään viitteitä, joiden avulla voidaan muodostaa taulujen rivien välille yhteys. Esimerkiksi jokaisella tuotetaulun rivillä on viite tuoteryhmätauluun. Yksi tuoteryhmä voi pitää sisällään yhden tai useamman tuotteen. Jos tuotteella on virheellinen tai tyhjä viite tuoteryhmään, linkkiä taulujen rivien välille ei voida tehdä (taulukon 3 rivi 4). Väärä voi tarkoittaa tyhjää arvoa, tai arvoa, jota ei ole olemassa. Jos tuoteryhmien avulla tehdään esimerkiksi tuoteryhmäkohtaisia myyntiarvioita, arviot menevät väärin tuotteiden osoittaessa tyhjiin tai vääriin tuoteryhmäriveihin.

ETL-prosessin virheilmoitukset riippuvat prosessin toteutustavasta (taulukon 3 rivi 5). Virheilmoitus kirjoitetaan esimerkiksi lokitiedostoon, tai se näytetään käyttöliittymässä. Hakuvaiheen virheilmoitus muodostetaan, kun datan hakeminen lähdejärjestelmästä ei ole onnistunut. Syy tähän on esimerkiksi virheellinen tietokantakysely lähdejärjestelmään, tai internetyhteyden katkeaminen. Muunnosvaiheen virheilmoitus muodostetaan, kun muunnettava data on jollain tavalla oletetusta poikkeavaa. Esimerkiksi jokin kenttä sisältää merkkijonoarvoja, kun sen oletetaan sisältävän vain lukuarvoja. Kirjoitusvaiheen virheilmoituksia ilmenee prosessoidun datan poiketessa oletetusta muodosta. Kirjoitusvaiheen virheilmoitus on esimerkiksi avainarvon puuttuminen datariviltä.

Uskottavuustarkistuksessa tarkistetaan, onko data järkevää (taulukon 3 rivi 6). Tietovarastoa käyttävä yritys tarkistaa uskottavuustarkistuksen aikana omista tiedoistaan, onko esimerkiksi myyntitapahtumia oikea määrä tiettyinä ajanjaksoina tietyissä toimipisteissä. Tarkistuksessa edetään tasoittain ylätason aggregaatiotasolta tarkemmalle tasolle. Esimerkiksi tarkistus aloitetaan tutkimalla myyntitapahtumien määrää yhteensä maatasolla ajanjakson aikana, jonka jälkeen tarkistetaan myyntitapahtumien määrää toimipistetasolla ajanjakson aikana ja lopulta tarkistetaan myyntitapahtumien määrää tuote-toimipistetasolla. Lisäksi tarkistusta tehdään pistokokeilla valitsemalla jonkun tuotteen toimipisteessä tiettynä päivänä ja tarkastamalla, onko myyntimäärä oikein kyseiselle päivälle.

Loppukäyttäjillä tarkoitetaan tietovarastohankkeen lopullisia käyttäjiä, eli henkilöitä, jotka käyttävät tietovarastossa olevaa dataa päivittäisessä työssään. Havaitessaan datavirheen loppukäyttäjät voivat ilmoittaa sen tietovarastoa tarjoavalle taholle, jolloin virhe voidaan selvittää ja korjata (taulukon 3 rivi 7).

Datan validointia voidaan pitää prosessina, joka koostuu kolmesta vaiheesta [23]. Prosessin päämääränä on havaita datavirheet ja parantaa datan laatua. Ensimmäisessä vaiheessa datasta etsitään poikkeavuuksia, jotka

heikentävät datan laatua. Seuraavaksi valitaan menetelmä, minkä avulla poikkeavuuksia voidaan löytää automaattisesti. Viimeisessä vaiheessa valittu menetelmä otetaan käyttöön, jonka seurauksena datan laatu voi parantua. Prosessiin on ehdotettu neljättä vaihetta, jossa tutkitaan, ovatko toimenpiteet tuottaneet haluttuja tuloksia [25].

4.2 Validointityökalujen ominaisuuksia

J. Bareteiron ja H. Galhardasin tekemässä kirjallisuuskatsauksessa käydään läpi erilaisia kaupallisia, sekä tutkimuspohjaisia datan validointiin tarkoitettuja työkaluja [6]. Tutkijat käyvät läpi 28 kaupallista, sekä yhdeksän tutkimuspohjaista validointityökalua ja niiden ominaisuuksia. Tutkijoiden kuvaamat ominaisuudet ovat lähinnä ETL-prosessiin kuuluvia ominaisuuksia. He listaavat seuraavat ominaisuudet yleisiksi:

- Tuki erilaisille *datalähteille* (engl. data sources). Datalähde voi olla esimerkiksi tietokanta tai tiedosto, kuten XML- tai CSV-tiedosto. Datalähteenä voi myös olla jokin ERP-järjestelmä.
- *Datan haku* (engl. extraction capabilities) datalähteestä on pystyttävä ajastamaan. Hakuihin on hyvä pystyä asettamaan erilaisia sääntöjä. Säännöt voivat liittyä tauluihin ja sarakkeisiin, tai sarakkeiden välisiin yhteyksiin.
- Työkalun on hyvä tukea kirjoitusoperaatioita (engl. loading capabilities) erilaisiin kohdejärjestelmiin. Kohdejärjestelmän dataa pitää pystyä lisäämään, päivittämään tai korjaamaan.
- Tuki *inkrementaaliseen päivitykseen* (engl. incremental updates). Kohdejärjestelmän dataa päivitetään tai korjataan inkrementaalisesti, eikä kaikkea olemassa olevaa dataa jokaisen suorituksen yhteydessä.
- *Käytettävyyden ja käyttöliittymän* (engl. interface) on oltava ymmärrettävä ja selkeää. Osa katsauksessa listatuista työkaluista tarjosi graafisen käyttöliittymän, joka teki tutkijoiden mukaan ohjelman käyttämisestä helpompaa.
- *Metadatan säilytys ja käyttö* (engl. metadata repository). Jotkin työkaluista säilyttävät käyttäjän muodostamat validointisäännöt tietokannassa tai erillisessä tiedostossa muistissa, sekä käyttävät näitä sääntöjä jokaisen suoritetun ajon yhteydessä. Metadatan [4] avulla voidaan pitää yllä erilaista tietoa datan sisältöön liittyen, kuten esimerkiksi kuinka suuri osa sarakkeista sisältää tyhjiä arvoja tai mikä sarakkeen datatyyppi on [30].
- *Suorituskyky ja skaalautuvuus* (engl. performance techniques and scala-

bility). Esimerkiksi säikeistämällä voi olla mahdollista parantaa suorituskykyä. Säikeistämisellä tarkoitetaan lähdekoodin jakamista osiin eli säikeisiin, jotka voidaan suorittaa samanaikaisesti usealla suorittimella. Työkalun pitää toimia tehokkaasti, että se skaalautuu suuriinkin datamassoihin.

- Työkalussa on hyvä olla joukko valmiita validointisääntöjä. Tutkijat käyttivät tästä joukosta nimeä *ominaisuuskirjasto* (engl. function library).
- Tuki *ohjelmointikielille* (engl. language binding). Halutessaan käyttäjä voi lisätä ominaisuuskirjastoon omia validointisääntöjä ohjelman tukemalla ohjelmointikielellä.
- Suorituksen *jäljitys ja seuranta* (engl. debugging and tracing). Suorituksen tapahtumia on hyvä pystyä seuraamaan, joko lokista tai graafisesta käyttöliittymästä. Käyttäjän on hyvä tietää esimerkiksi haetun, korjatun ja kirjoitetun datan rivimäärät.
- *Poikkeusten huomaaminen ja hallinta* (engl. exception detection and handling). Poikkeavia tapahtumia on hyvä pystyä seuraamaan, joko graafisesta käyttöliittymästä, erillisestä lokitiedostosta tai ulkoisesta tietokannasta. Poikkeavia tapahtumia ovat esimerkiksi syötteet, jotka jostain syystä saivat työkalun poikkeustilaan. Esimerkiksi tyhjät arvot tai väärä datatyyppi voivat aiheuttaa poikkeuksen.
- *Datan alkuperä* (engl. data lineage). Työkalun on hyvä ilmaista korjatun datan alkuperä eli mistä tai miten tieto on päätynyt tietokantaan. Esimerkiksi datan alkuperä voi olla lähdejärjestelmä tai käyttäjän antama syöte. Jos alkuperä on selvillä, on helpompaa tehdä jatkotutkimuksia.

Yllä olevien ominaisuuksien lisäksi useat tutkimukset kertovat työkaluista, jotka ovat erikoistuneet vain tiettyyn validoinnin osa-alueeseen [30, 25]. Esimerkiksi IntelliClean-työkalu on erikoistunut duplikaattiarvojen validointiin [25].

5 Tapaustutkimus: Relex

Tutkielman empiirisessä osassa määritellään ja toteutetaan työkalu case-yritykselle. Työkalun tarkoituksena on automatisoida datan validointia case-yrityksen projekteissa ja tällä tavalla nopeuttaa validointiprosessia. Luvussa 5.1 esitellään case-yritys ja luvussa 5.2 tehdään validointityökalun vaatimusmäärittely.

Tutkimuksen tekijä on töissä yrityksessä nimeltä Relex. Tutkimus tehdään tapaustutkimuksena, jossa aineistoa kerätään teemahaastatteluilla ja havainnoimalla käyttäen apuna ”ajattele ääneen”-menetelmää [33].

5.1 Tapaustutkimus tutkimusmenetelmänä

Tapaustutkimukset soveltuvat tietojenkäsittelyyn, sillä tapaustutkimukset tutkivat nykyaikaisia ilmiöitä luonnollisessa ympäristössään [33]. Tapaustutkimukseen tekoon liittyy viisi vaihetta, jotka on käytävä läpi tutkimusta tehdessä [33]. Näistä ensimmäinen vaihe on tapaustutkimuksen suunnitelman muodostaminen (engl. case study design), missä määritellään tutkimuksen päämäärä. Toisessa vaiheessa muodostetaan datan keräyssuunnitelma (engl. preparation for data collection), jonka avulla tapaustutkimuksessa kerätään dataa. Seuraavassa vaiheessa todisteet kootaan (engl. collecting evidence) käyttämällä toisessa vaiheessa määriteltyä keräyssuunnitelmaa. Neljännessä vaiheessa kerätty data analysoidaan (engl. analysis of collected data) ja lopulta viidennessä vaiheessa analyysin tulokset raportoidaan (engl. reporting).

Tutkimuksilla voi olla useita eri päämääriä. Tutkimukset voidaan jaotella päämäärien perusteella neljään eri kategoriaan: tutkivaan (engl. exploratory), kuvailevaan (engl. descriptive), selittävään (engl. explanatory) ja kehittävään (engl. improving) tutkimukseen [32]. Tutkivissa tutkimuksissa muodostetaan uusia näkemyksiä ja hypoteeseja uusille tutkimuksille. Kuvailevassa tutkimuksessa pyritään esittämään jokin ilmiö. Selittävässä tutkimuksessa etsitään vastausta johonkin ongelmaan tai tilanteeseen. Kehittävässä tutkimuksessa pyritään parantamaan tutkittavaa ilmiötä.

Kvalitatiivista dataa voidaan kerätä eri keinoin, kuten havainnoimalla osallistujaa (engl. participant observation) tai haastattelemalla. Kyseiset keinot ovat hyödyllisiä keräämään ensikäden tietoa tietojenkäsittelytieteessä [35]. Osallistujan havainnoinnissa on ideana olla mukana seuraamassa ja dokumentoimassa osallistujaa tekemässä jotain aktiviteettia [35]. Osallistuja tietää olevansa havainnoinnin kohteena. Tietojenkäsittelyssä havainnointia voi olla haastavaa tehdä, sillä esimerkiksi ohjelmistokehitys tapahtuu suurelta osin kehittäjän pään sisällä [35]. Ajattelua on vaikeaa havainnoida, mutta apuna voi käyttää ”ajattele ääneen”-menetelmää, missä osallistujaa pyydetään kertomaan ja kuvaamaan ajatuksiaan ääneen, jolloin tutkija oppii ja ymmärtää tutkittavasta aktiviteetista [33]. Osallistujan havainnoinnissa on otettava huomioon asioita, jotka voivat vääristää havainnoinnista saatua dataa. Osallistujien ei tulisi ottaa paineita paikalla olevasta tutkijasta, vaan tilanne pitäisi pyrkiä muodostamaan ja pitämään mahdollisimman luonnollisena [35].

Toinen tietojenkäsittelyssä yleisesti käytössä oleva keino kerätä ensikäden tietoa on haastattelut [35]. Haastattelut voidaan jakaa strukturoituihin, puolistrukturoituihin ja strukturoimattomiin haastatteluihin [35]. Strukturoitu haastattelu on lomakekysely tai lomakehaastattelu, missä tutkimukseen vastanneet henkilöt voidaan tyypitellä vastaustensa perusteella laadullisiin luokkiin [40]. Puolistrukturoidussa haastattelussa eli teemahaastattelussa pysytään etukäteen valitun aihealueen ympärillä [40]. Haastattelua varten on mietitty joukko kysymyksiä, mitä voidaan käyttää haastattelun runkona,

mutta haastateltavan vastausten perusteella kysymyksiä voidaan syventää ja tarkentaa [40]. Strukturoimaton eli syvähaastattelu on avoin haastattelu, mihin on valittu ainoastaan ilmiö, josta keskustellaan [40].

5.2 Tutkimussuunnitelma

Tapaustutkimuksen tekoon liittyy viisi vaihetta: suunnitelman muodostaminen, datan keräyssuunnitelma, todisteiden koonti, datan analysointi ja raportointi [33]. Seuraavaksi esitellään kyseiset vaiheet tähän tutkimukseen liittyen.

Tämä tutkimus on tutkiva tapaustutkimus, minkä päämääränä on toteuttaa case-yrityksen tietovarastohankkeiden työntekijöille työkalu, joka nopeuttaa datan validointiprosessia raportoimalla case-yrityksen asiakkaiden tietovarastojen sisältämästä virheellisestä datasta. Jotta päämäärään päästään, tutkielman tekijän on ymmärrettävä, mitä virheellinen data tarkoittaa case-yrityksen näkökulmasta ja mitkä ovat asiakashankkeissa yleisesti toistuvat dataongelmat. Tämän jälkeen tutkielman tekijä pyrkii muodostamaan case-yritykselle soveltuvan työkalun ongelman ratkaisemiseksi.

Koska tutkimuksen päämääränä on muodostaa työkalu, työkalun käyttötapauksista kerättiin aineistoa. Työkalun käyttötapaukset (jatkossa validointitapaukset) muodostettiin poimimalla aihealueeseen liittyvistä tutkimuksista yleisiä datan laatuongelmia [30, 6, 16] ja täydennettiin haastattelemalla yrityksen asiantuntijoita. Lisäksi tutkielmasta tiedotettiin case-yrityksen kommunikointipalvelussa, jonne case-yrityksen työntekijät lähettivät datan laatuongelmiin liittyviä dokumentteja.

Tutkimukseen liittyviin haastatteluihin osallistui seitsemän henkilöä. Haastatteluihin pyydettiin henkilöitä, joilla oli kokemusta useista tietovarastohankkeista. Haastatteluihin osallistui tuotekehityksen johtoryhmään kuuluva henkilö, neljä projektipäällikköä, yksi aluejohtaja ja yksi tekninen projektipäällikkö. Haastattelut pidettiin kasvotusten case-yrityksen toimistolla. Haastattelut kestivät noin tunnin ajan. Haastattelut olivat teemahaastatteluita, sillä haastatteluissa haluttiin pysyä datan laatuun ja validointiin liittyvässä aihealueessa ja tietoa haluttiin kerätä case-yrityksen sisällä aiheesta hyvin perillä olevilta henkilöiltä. Haastatteluissa käytettiin kysymyslistaa haastattelujen runkona, mutta valmisteltujen kysymysten lisäksi täsmentäviä kysymyksiä kysyttiin tarvittaessa. Haastattelut nauhoitettiin ja tutkielman tekijä kirjoitti muistiinpanoja haastattelujen aikana. Haastattelujen lisäksi tutkielman tekijä keräsi teknisiltä projektipäälliköiltä tietoa tyypillisistä dataongelmista case-yrityksen tietovarastohankkeissa vapaasti keskustelemalla. Viimeisten haastattelujen aikana tutkielman tekijä huomasi, ettei uusia validointitapauksia enää ilmennyt, joten uusien haastattelujen pitämistä ei pidetty tarpeellisena. Haastattelut käytiin läpi lukemalla haastattelujen aikana käydyt muistiinpanot, sekä kuuntelemalla nauhoitteita.

Ohjelmiston vaatimuksia voidaan kartoittaa tutkimalla käyttäjiä tekemäs-

sä heidän työtään [37]. Asiantuntijoita pyydettiin näyttämään ja kuvailemaan, kuinka he validoivat dataa case-yrityksen ohjelmistoa käyttäen. Tutkielman tekijä pystyi havainnoimaan haastateltavaa validoimassa dataa. Tutkielman tekijä pyrki käyttämään niin sanottua ”ajattele ääneen”-menetelmää, jolloin tutkielman tekijällä oli mahdollisuus nähdä ja oppia validointityötä käytännössä sekä tehdä havaintoja, miten validointiin liittyviä prosesseja olisi mahdollista automatisoida.

Tutkimuksessa pyritään osoittamaan, että tutkielmassa toteutettu työkalu toimii case-yrityksessä, joka vastaa vaihetta ”todisteiden koonti”. Työkalun ominaisuuksia verrataan aiemmissa tutkimuksissa muodostettuihin työkaluihin luvussa 7.1. Lisäksi työkalun toimintaa arvioidaan luvussa 7.2 suorittamalla se case-yrityksen tuotteen asiakasasennuksissa ja kokoamalla tuloksia. Tämän jälkeen luvussa 8.1 tutkielmassa tehdyt löydökset analysoidaan ja raportoidaan.

5.3 Validointityökalun vaatimusmäärittely

Työkalun tarkoituksena on karsia datan validointiin käytettävää aikaa etsimällä projekteissa toistuvia datavirheitä tietokannasta automaattisesti. Virheistä muodostetaan raportti, jonka avulla niihin voidaan puuttua yhdessä Relexin asiakkaiden kanssa. Raportin avulla virheet pyritään havaitsemaan aiempaa varhaisemmin ja niitä voidaan karsia. Tällöin työkalun avulla pyritään nopeuttamaan datan validointia. Työkalua on tarkoitus käyttää ETL-prosessin kirjoitusvaiheen jälkeen.

Työkalun tulee toimia jokaisessa case-yrityksen asiakasasennuksessa riippumatta asiakkaan lähettämän datan muodosta. Lähtökohtaisesti työkaluun ei tehdä ominaisuuksia, jotka toimivat vain yksittäisissä case-yrityksen asiakkaille tehdyissä tietovarastoasennuksissa. Case-yrityksen tuote perustuu tietokantaan, joka on toteutettu yrityksen sisällä. Työkalun toiminta perustuu tietokantakyselyihin, joten sen avulla voidaan validoida ainoastaan case-yrityksen tietokannassa olevaa dataa.

Työkalun asiakkaina toimivat case-yrityksen tietovarastohankkeiden projektitiimiläiset, jotka voivat olla joko teknisiä henkilöitä, tai vähemmän teknisiä henkilöitä. Työkalun käyttö pyritään suunnittelemaan helpoksi, jolloin sen käyttö ei vaadi teknistä osaamista. Työkalun raportointi halutaan muodostaa Excel-formaattiin ja sen tulee olla helposti ymmärrettävää ja yksiselitteistä.

Seuraavat validointitapaukset kiteytettiin aiempien tutkimusten ja haastattelujen pohjalta:

- Myyntien puuttuminen toimipisteissä kuukausi/viikkotasolla
- Myyntien puuttuminen tuoteryhmissä kuukausi/viikkotasolla
- Negatiiviset saldot transaktiotasolla

- Tuote- ja toimipisteviitekoodien virheettömyys
- Yli kaksi viikkoa myöhässä olevien toimitusten määrä
- Tuotteiden tyhjien ryhmäviitteiden määrä
- Tuote-lokaatio datan viitteiden virheettömyys
- Ylimääräisen datan etsiminen tuoteryhmä-, tuote-, toimipiste- ja tuote-toimipiste -tasolla
- Automaattiset validointinäkymät
- Duplikaattitarkistukset

Validointitapaukset esitellään tarkemmin tutkielman luvussa 6.5 esimerkkiraportin avulla.

5.4 Kohdeyrityksen esittely

Case-yritys on toimitusketjuun erikoistunut yritys, jonka päätuote on Relex niminen ohjelmisto. Toimitusketju tarkoittaa kokonaisuutta, missä tuotteet liikkuvat raaka-ainevaiheesta lopullisille asiakkaille saakka. Relex-ohjelmisto muodostaa myyntiennusteita asiakkaiden toimittamien historiallisten myyntien perusteella. Ohjelmistolla on mahdollista generoida tuotteiden tilausehdotuksia, jotka pohjautuvat myyntiennusteisiin. Tilausehdotukset kertovat kuinka paljon tuotetta kannattaa määrällisesti tilata tiettynä päivänä tiettyyn toimipisteeseen eli lokaatioon. Ohjelmisto perustuu liiketoiminnan kuvaamiseen tarkoitettuun dataan, joka jaotellaan Relexissä informatiiviseen, niin kutsuttuun *masterdataan*, sekä aikariippuvaiseen dataan, josta käytetään nimitystä *transaktiodata*. Relex tarjoaa monelle sen asiakkaille tietovarasto-ohjelmistoa, jonka avulla yritykset pystyvät näkemään kokonaiskuvan liiketoiminnastaan.

Masterdata on informatiivista tietoa, joka on esimerkiksi tietoa yrityksen tuotteista, lokaatioista, kampanjoista tai tuoteryhmistä. Kaikki masterdata kirjoitetaan Relexin tietokantaan tiedostoista. Tiedoston nimi kertoo, mihin tietokantatauluun tiedoston sisältö kirjoitetaan.

Relexissä suunnittelua voi tehdä eri tasoilla. Tarkin taso on tuote-lokaatiotaso, joka tarkoittaa yhtä tuotetta yhdessä lokaatiossa. Tämä ilmaistaan muodossa "tuote / lokaatio", esimerkiksi "Koff III, tölkki / S-market Sörnäinen". Tuote-lokaatio -taso on Relexin toiminnan kannalta tärkein taso, sillä suuri osa Relexissä muodostettavasta ohjausdatasta lasketaan tuote-lokaatiotasolla. Ohjausdata sisältää muun muassa automaattisesti muodostettavat ennusteet tai tilausehdotukset.

Relexin liiketoimintalogiikka perustuu asiakkaiden transaktiodataan, jota on muun muassa myyntitapahtumat, saldot sekä tilaukset. Kaikki transaktiot

kirjoitetaan tuote-lokaatio-päivä -tasolle eli tuote-lokaation myyntitapahtuma liittyy aina tiettyyn päivään. Transaktioiden aikasidonnaisuuden avulla Relex laskee tuotteiden kysyntää eri lokaatioissa eri ajankohtina. Käytössä olevia ennustemalleja on useita, kuten esimerkiksi sesonkimalli [15]. Ennustemallien valinta on tuote-lokaatiokohtaista. Relex päättelee käytettävän ennustemallin automaattisesti saadun datan perusteella, mutta käyttäjän on mahdollista asettaa se manuaalisesti. Saadun datan ja siitä tehtävän ennustelaskennan perusteella Relexiä käyttävien tahojen on mahdollista tehdä tarkkaa tilausten-, myynnin- ja tuotannonsuunnittelua.

Relexin käyttöönottoprojektia voidaan verrata muihin tietovarastojen käyttöönottoprojekteihin. Relexin käyttöönottoprojektin aikaavievin osa-alue on datan muovaaminen laadukkaaksi, joka on tutkimusten mukaan yleisesti aikaavievin osa-alue tietovarastohankkeissa [16]. Relexiin halutaan rakentaa työkalu, joka etsii sekä skeema- että instanssitason datavirheitä. Tämän hetkinen ETL-prosessi estää kuitenkin useat yleiset skeematason virheet. Esimerkiksi viite-eheys-, datatyyppi-, sekä duplikaattitarkistuksia on jo olemassa. Tämän takia suuri osa tarkistuksista ovat instanssitasoisia eli liittyvät datan sisältöön.

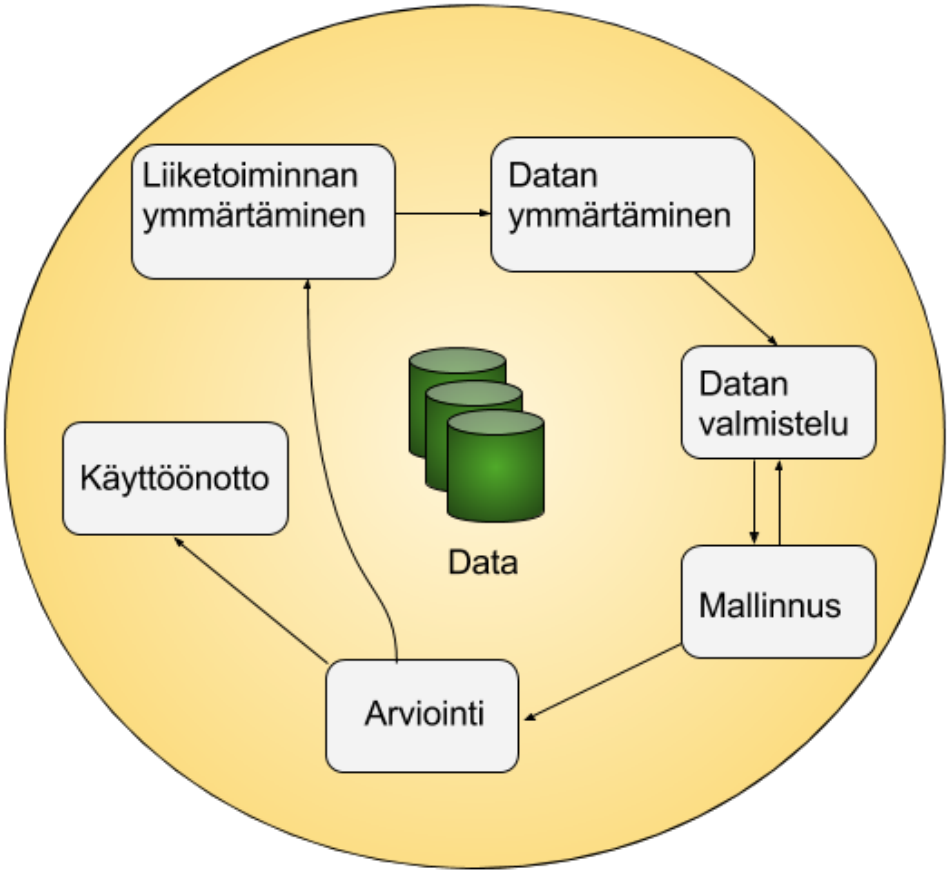
6 Työkalun toteutus

Tässä luvussa esitellään, millä tavalla työkalu kehitettiin käymällä läpi työkalun kehitystä varten muodostettu kehitysmalli. Luvussa 6.3 kerrotaan työkalun arkkitehtuurista käymällä läpi työkalun UML-kaavio, sekä työkaluun liittyvä arkkitehtuurimalli. Luvussa 6.4 kerrotaan ohjelmointiesimerkein työkalun suoritukseen liittyviä tärkeimpiä luokkia. Työkalu muodostaa suorituksen päätteeksi raportin, joka esitellään luvussa 6.5.

6.1 Työkalun lähtökohdat ja suunnitelma

Tällä hetkellä tietojenkäsittelyn alalla yleisesti suosituin ohjelmistokehityksen muoto on ketterä ohjelmistokehitys (engl. agile software development). Työkalun toteuttamiseen käytettiin soveltaen Scrum-ohjelmistokehitysmallia [21], missä asiakkaana toimii case-yrityksen työntekijät. Kehitettävät ominaisuudet kirjoitetaan case-yrityksen sisäiseen kehitykseen tarkoitettuun sovellukseen, jonka avulla voidaan ylläpitää kehitysjonoa (engl. backlog), sekä sprintin tehtävälistaa (engl. sprint backlog). Tutkielman tekijä toteutti kaikki tehtävät ja tarvittaessa konsultoi ja miettii työkalun ominaisuuksien teknisiä vaatimuksia case-yrityksen vanhempien ohjelmoijien kanssa. Case-yrityksen vanhemmat ohjelmoijat tekevät koodin laadunvarmistamiseksi koodikatselmointia jokaisesta toteutetusta ominaisuudesta. Tutkielman tekijän on mahdollista saada nopeasti ja ketterästi palautetta asiakkaalta käymällä ominaisuuksia läpi aina uuden ominaisuuden valmistuessa. Prosessi pidetään syklisenä, missä

syklien pituudet ovat noin viikon mittaisia. Ominaisuuksia kehitetään niin kauan kuin jatkokehitys nähdään tarpeellisena ja työkalua eteenpäin vievänä.



Kuva 7: Tiedonlouhintamalli [46]

Työkalun suunnitteluun ja toteutukseen käytettiin apuna liiketoiminnalliseen tiedonlouhintaan tehtyä mallia ks. kuva 7 [46]. Tiedonlouhinta on tietojenkäsittelyn ala, jossa suuria määriä dataa jäsennellään muotoon, jossa sitä voidaan helpommin tutkia ja käyttää myöhemmässä analyysissä [7]. Tiedonlouhinnan pääpiirteet ovat suuret datamäärät tai tietokannat, joita käsitellään erilaisten algoritmien tai tekoälyjen avulla.

Kuvan kohdassa ”Liiketoiminnan ymmärtäminen” pyritään muuttamaan liiketoiminnallinen ongelma tiedonlouhintaan liittyväksi ongelmaksi. Työkalun kehityksen kannalta tämä tarkoittaa case-yrityksen tuotteen syväanalyysiä, jossa selvitetään mitä tuotteen ominaisuuksia käytetään lähes kaikissa asiakkaille tehdyissä asennuksissa. Tästä ominaisuusjoukosta etsitään ominaisuudet, jotka tekevät laskentaa datan perusteella. Liiketoiminnan ymmärtämiseen on vahvasti sidoksissa kohta ”Datan ymmärtäminen”. Työkalukehityksessä tämä tarkoittaa yhteyttä case-yrityksen tuotteen ja

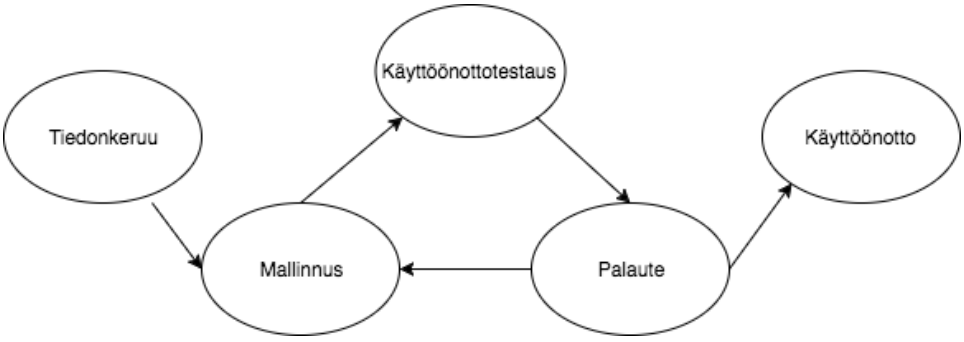
tietokantaan kirjoitettavan datan välillä.

Kohdassa ”Datan valmistelu” dataa suodatetaan ja siitä etsitään kiinnostavia tekijöitä. Työkalukehityksessä tämä tarkoittaa, että esimerkiksi transaktiodatasta pyritään löytämään aikajaksoja, milloin jossain myymälässä ei ole ollut lainkaan myyntiä. Myyntiaukot tarkoittavat usein datavirhettä. Mallinnusvaiheessa määritellään, millä parametreilla kiinnostavaa dataa etsitään. Esimerkiksi jos jossain myymälässä ei ole ollut myyntiä yhtenä päivänä, onko kyseessä aina datavirhe? Monet myymälät eivät ole auki esimerkiksi sunnuntaisin tai pyhäpäivinä, jolloin päivätasoinen tarkastelu ei ole järkevää.

Näiden askelien jälkeen on muodostettu malli, jonka avulla voidaan louhia isosta datamassasta kiinnostavia piirteitä. ”Arviointi”-vaiheessa mietitään, onko malli toimiva. Mallia testataan ja tarkastetaan täyttääkö se vaatimukset ja otetaanko reunatapaukset huomioon. Käyttöönottovaiheessa analyysin tulokset voidaan kirjata muotoon, missä asiakas niitä ymmärtää. Lisäksi voidaan kirjoittaa tarkempi prosessi, kuinka tehdä tiedonlouhintamalleja jatkossa.

6.2 Kehitysmalli

Soveltaen tiedonlouhintamallia artikkelista [46] muodostettiin kuvan 8 mukainen kehitysmalli tukemaan työkalun kehitystyötä. Kehitys alkaa tiedonkeruulla. Tiedonkeruuta tehdään tutkimalla aiheeseen liittyvää tutkimusta, sekä haastattelemalla case-yrityksen asiantuntijoita. Tiedonkeruun yhteydessä pyritään tekemään erilaisia validointitapauksia. Validointitapaus tarkoittaa esimerkiksi myyntidatan puuttumista lokaatiossa jonkin kuukauden aikana. Tapaukset lisätään työkalun kehitysjonoon.



Kuva 8: Työkalun kehityksen avuksi muodostettu malli artikkelia [46] mukaillen.

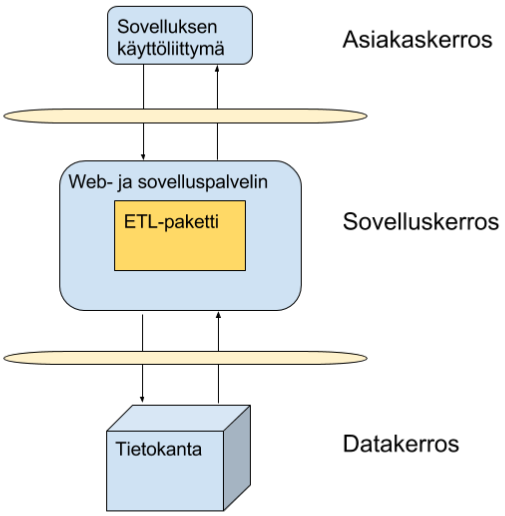
Mallinnusvaiheessa validointitapaukset ohjelmoidaan yksitellen osaksi työkalua. Ennen ohjelmoinnin aloittamista selvitetään validointitapauksen kaikki tekniset tarpeet. Teknisiin tarpeisiin kuuluvat esimerkiksi ohjelmistoarkkitehtuurillinen suunnittelutyö sekä validointitapauksen muokkaami-

nen tietokantakyselyksi. Mallinnusvaiheessa validointitapaukset testataan yksikkötestien avulla.

Käyttöönottotestauksessa validointitapaus viedään testiympäristöön testattavaksi oikealla asiakkaan testidatalla. Testiympäristö on Relex-ohjelmiston instanssi, mitä käytetään pääasiassa uusien ominaisuuksien testaamiseen. Työkalu muodostaa tietokantakyselyt ja kirjoittaa kyselyjen vastausten perusteella raportin. Tulokset tarkistetaan vertailemalla tietokannan tilaa raportin sisältöön. Validointitapauksesta pyydetään palautetta asiakkaan testiympäristöstä vastaavalta henkilökunnalta. Kehitystä jatketaan kunnes uusia ideoita tai ajatuksia ei enää synny. Tämän jälkeen validointitapaus viedään lopullisesti osaksi validointityökalua. Uuden validointitapauksen käyttöönoton jälkeen prosessi aloitetaan alusta keräämällä tietoa, tai mallintamalla seuraavana kehitysjonossa olevaa validointitapausta.

6.3 Arkkitehtuuri

Tätä työtä voidaan kuvata asiakas-palvelin -tyyliin pohjautuvalla kolmitasoarkkitehtuuri-mallilla, joka esitellään kuvassa 9. Työkalu toteutettiin Java-ohjelmointikielellä, koska case-yrityksen tuote on suurimmalta osin Javaa. Työkalu toteutettiin irralliseksi moduuliksi case-yrityksen tuotteen ETL-pakettia. Case-yrityksessä ETL-paketti on erillinen osa itse tuotetta.

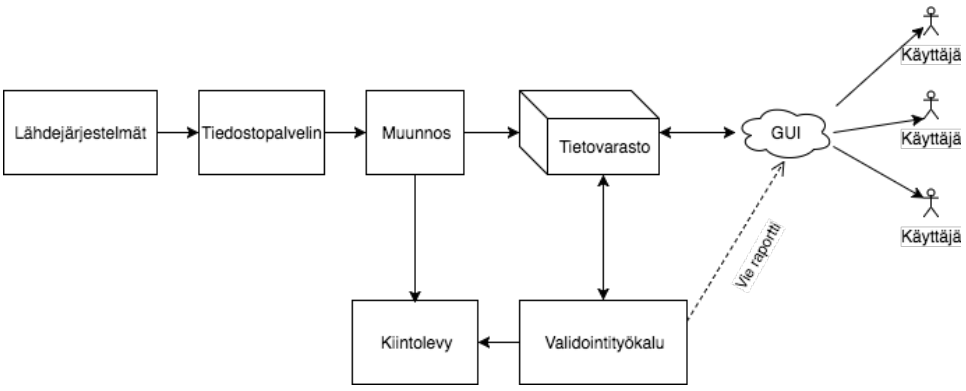


Kuva 9: Kolmitasoarkkitehtuuri-malli

Työkalun arkkitehtuuriin liittyy case-yrityksen ETL-prosessi, mikä esitellään kuvassa 10. Case-yrityksen ETL-prosessia voidaan pitää vertailukelpoisena tutkimuksessa [10] esiteltyyn prosessiin, josta kerrottiin luvussa 2.2. Suurin ero on väliaikaiskannan puuttuminen Relexissä. Kuvassa 3 muunnosohjelmat tallentavat dataa väliaikaiskantaan sekä voivat lähettää dataa suoraan tietovarastoon. Relexin ETL-prosessissa muunnosfunktiot jäsentävät datan haluttuun muotoon, jonka jälkeen data on valmiina kirjoitettavaksi tietovarastoon.

Työkalun toiminta perustuu tietokantakyselyihin. Jokainen validointitapaus ohjelmoitiin erilliseksi luokaksi, joka tekee ainakin yhden tietokantakyselyn. Vastauksen perusteella muodostetaan päättely, onko tietokannassa validia dataa vai ei. Jokainen validointitapaus kirjoittaa vastaukset excel-raporttiin, jonka käyttäjä voi ladata koneelleen Relex-ohjelmiston käyttöliittymästä.

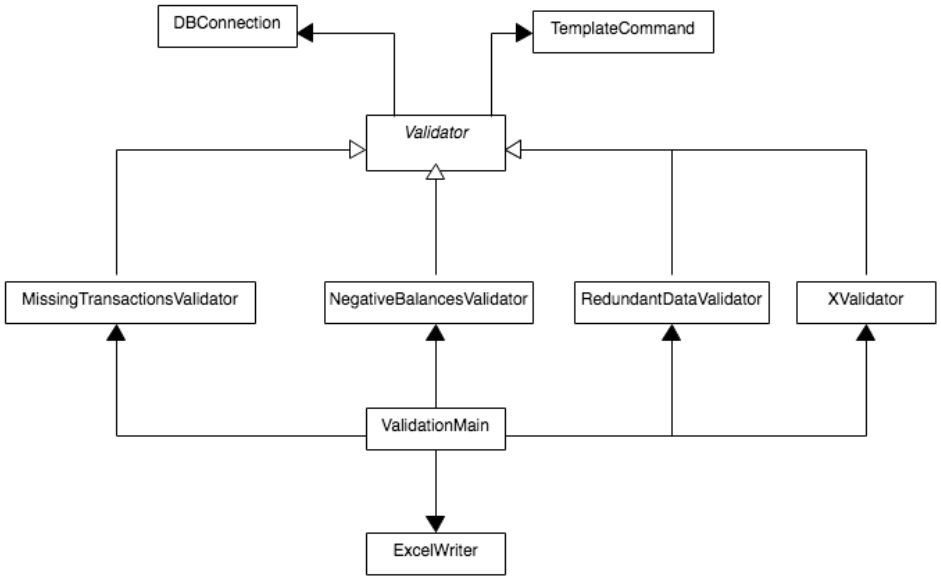
Työkalun käyttö tapahtuu pilvipalveluna toimivan Relex-ohjelmiston kautta. Käyttöliittymästä voidaan aloittaa työkalun suoritus nappia painamalla. Napin painalluksesta sovellus lähettää HTTP-pyynnön palvelimelle, joka käynnistää palvelimella (sovelluskerros) olevan työkalun. Tämän jälkeen työkalu aloittaa lähettämään validointikyselyitä HTTP:n yli tietokantaan (datakerros). Tietokannasta tulevien vastauksien perusteella työkalu rakentaa raportin, tallentaa sen palvelimelle sekä muodostaa käyttäjälle latausikkunan, josta raportin voi ladata. Raportti kuvaa tietokannassa olevan datan oikeellisuutta eli validiteettia.



Kuva 10: ETL-prosessi Relexissä.

Kuvassa 11 esitettynä validointityökalun luokkakaavio. Abstraktia Validator-luokkaa laajentavia aliluokkia on kahdeksan. Yksinkertaistuksen vuoksi kuvaan 11 ei piirretty kuin neljä aliluokkaa. Kuvan 11 kohdalla *XValidator* kuvataan mallintamatta jätettyjä aliluokkia. Jokainen validointitapaus eli *Validator*-luokkaa laajentava luokka esitellään tutkielman myöhemmässä vaiheessa. Kuvassa olevan luokan *TemplateCommand* avulla rakennetaan tietokantakyselyitä. Luokan logiikka esitellään myöhemmin esimerkin avulla.

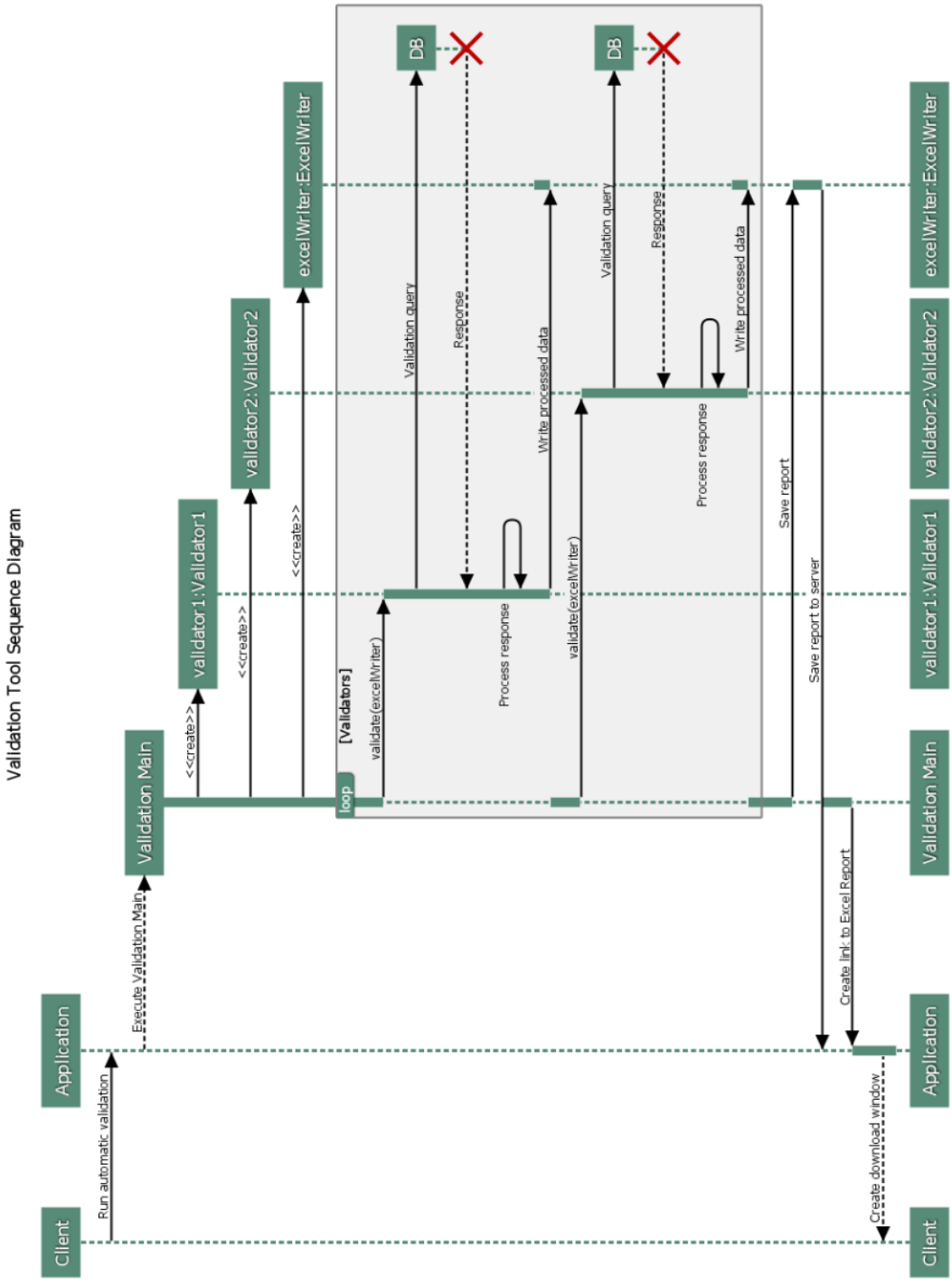
DBCOnnection-luokan avulla muodostetaan yhteys tietovaraston tietokantaan.



Kuva 11: Työkalun luokkakaavio

Kuvassa 12 esitellään työkalun käyttöprosessi sekvenssikaaviona. Kuvassa olevat validointi-instanssit (validator1, validator2) mallintavat validointitapauksia. Luokkien nimet kuvaavat, mikä luokan tarkoitus on. Esimerkiksi negatiivisia saldoja etsivän validointiluokan nimi *NegativeBalancesValidator*. Kaikkien luokkien suoritus on hyvin lähellä toisiaan, minkä takia luokkien nimet ovat merkattu kaavioon yleistetyssä muodossa. Kaavioon on merkattu samaisesta syystä vain kaksi validointi-instanssia, vaikka työkalu suorittaa 12 validointi-instanssia ajaessaan jokaisen validointitapauksen. Suoritettavien validointitapausten määrää voi rajoittaa parametrien avulla.

ValidationMain-luokka toimii työkalun pääluokkana. Kyseinen luokka alustaa jokaisen ajoon tarvittavan luokan eli Excel-kirjoittimen sekä suoritettavat validointiluokat. Kaikki validointiluokat perivät abstraktin luokan *Validator*. *Validator*-luokkaan on muodostettu abstrakti metodi *validate*, jolloin jokainen luokka, joka perii *Validator*-luokan toteuttaa metodin *validate*. Kaikkien validointiluokkien *validate*-metodi suoritetaan yhdessä toistorakenteessa, jossa käydään validointiluokkien lista läpi. *ValidationMain*-luokka antaa validoinnin aikana jokaiselle validointitapaukselle Excel-kirjoittimen parametrina kutsuessaan *validate*-metodia. Validointiluokka antaa Excel-kirjoittimelle validoinnin tulokset tietokantavastausten ja niiden prosessoinnin jälkeen. Excel-kirjoittimeen luotiin useita eri metodeja eri käyttötarkoituksia varten. Jokainen validointitapaus kirjoittaa yhteenvetosivulle löydöksensä metodissa nimeltä *createSummaryPageValues*. Osa luokista kirjoittaa erilliselle välileh-



Kuva 12: Automaattisen validoinnin suoritusta kuvaava sekvenssikaavio (yksinkertaistettu)

delle tarkemman kuvauksen löydöksistä. Nämä luokat ylikirjoittavat metodin nimeltä *createExcelReport*. Excel-raportin sisältöä käydään läpi tarkemmin tutkielman myöhemmässä vaiheessa.

Työkalun suoritus tapahtuu asynkronisesti suhteessa case-yrityksen sovellukseen, mikä tarkoittaa työkalun suorituksen olevan riippumaton sovelluksen tilasta. Käyttäjä voi siis jatkaa web-sovelluksen käyttöä validointityökalun suorituksen aikana normaalisti. Käyttäjän on mahdollista päättää, mitä validointitapauksia työkalu suorittaa antamalla työkalulle parametreja. Käyttäjän on mahdollista aloittaa ja keskeyttää työkalun suoritus. Kun työkalu on edennyt suorituksensa loppuun, web-sovelluksen käyttöliittymään ilmestyy latausikkuna, mistä käyttäjä voi ladata Excel-raportin. Excel raportin sisällöstä kerrotaan tarkemmin luvussa 6.4.

6.4 Suorituksen perusperiaatteet

Työkalun suoritus on synkroninen tapahtuma. Kaikki validointi-instanssit ovat tallessa tietorakenteessa ja ne suoritetaan peräkkäin. Kuvan 12 validointi-instansseista on karsittu yhteys luokkaan *TemplateUtil*, jonka avulla tietokantakyselyt muodostetaan. Tietokantakyselyt muodostetaan suorituksen aikana. Tekniikka esitellään ohjelmointiesimerkissä 1.

Ohjelmointiesimerkki 1.

```
1  package fi.relex.validation;
2
3  import fi.relex.customer.util.DateUtil;
4  import fi.relex.customer.util.FastormQuery;
5  import fi.relex.validation.util.TemplateUtil;
6
7  import java.util.HashMap;
8  import java.util.Map;
9
10 public class TemplateExample {
11
12     public static final String CMD_EMPTY_SALES_ON_TIME_PERIOD =
           "begin_trx.cube.slice(Fastorm::LocalDate.new('{:dateFrom}'),
           Fastorm::LocalDate.new('{:dateTo}')).slice(\"{:table}\",
           \"sum(transactions.sales_quantity) =
           null\").view(\"{:table}\").each{|row| puts \"#{row.code} /
           #{row.name}\"};";
13
14     public static void main(String[] args) throws Exception {
15         TemplateUtil templateUtil = new
               TemplateUtil(CMD_EMPTY_SALES_ON_TIME_PERIOD);
16         Map<String, String> options = new HashMap<>();
17         options.put("{:dateFrom}", DateUtil.getDateMinus(7));
18         options.put("{:dateTo}", DateUtil.getToday());
19         options.put("{:table}", "locations");
20         String query = templateUtil.replace(options);
21         System.out.println(query);
22         //Prints:
               begin_trx.cube.slice(Fastorm::LocalDate.new('2017-03-21'),
               Fastorm::LocalDate.new('2017-03-28')).slice("locations",
               "sum(transactions.sales_quantity) =
               null").view("locations").each{|row| puts "#{row.code} /
               #{row.name}"};
23         FastormQuery.runRunner(query);
24         //Response: list of locations without sales during last week
25     }
26 }
```

Työkaluun ohjelmoitiin keino muodostaa tietokantakyselyitä, jotka sisältävät muuttujia. Monessa validointitapauksessa tietokantakyselyt muodostetaan ohjelman suorituksen aikana, sillä ne ovat riippuvaisia suorituksen aloitusajankohdasta. On useita tapoja täyttää tämä vaatimus, kuten esimerkiksi Javan StringBuilder [28]. Kehityksen aikana päädyttiin soveltamaan web-kehityksestä tuttua malleihin (engl. template) perustuvaa tekniikkaa. Malleja käyttävä luokka *TemplateUtil* kirjoitettiin työkalua varten osaksi case-yrityksen ETL-pakettia. Ohjelmointiesimerkissä 1 rivillä 15 luodaan *TemplateUtil*-olio, joka saa konstruktorin parametrina tietokantakyselymal-

lin.

Case-yritys käyttää omiin tarpeisiinsa mukautettua tietokantaa. Mallikysely on rivillä 12 esimerkissä 1 merkkijonomuuttujassa *CMD_EMPTY_SALES_ON_TIME_PERIOD*. Kyselyn avulla voi etsiä mistä tahansa taulusta, sekä miltä tahansa ajanjaksolta puuttuvia myyntejä. Esimerkissä kysely kohdistuu tauluun *locations*. Aikajaksona käytetään viime viikkoa. Kysely palauttaa listan toimipisteitä, jotka eivät myyneet mitään viime viikon aikana.

Ohjelmointiesimerkissä 2 on tuotu esille myyntitapahtumia tutkivan *TransactionsValidator*-luokan toimintaperiaatteita alkaen riviltä 7. Metodi *run* sisältää luokan pääasiallisen logiikan. Rivillä yhdeksän ja kymmenen muodostetaan mallikysely ja mallikyselyitä käyttävä *TemplateUtil*-olio.

Rivillä 13 aloitetaan käymään läpi *timePeriods*-listaa, joka sisältää tarkasteltavan aikajakson kaikki päivämäärät. Jos haluamme tutkia kuukauden aikajaksoa viikkotasolla alkaen päivästä 2017-01-01, lista sisältää päivämäärät "2017-01-02", "2017-01-08", "2017-01-09", "2017-01-15", "2017-01-16", "2017-01-22", "2017-01-23", "2017-01-29". Työkalu käy aina läpi täysiä viikkoja, jotka alkavat maanantaista. Tämän takia listan ensimmäisenä päivämääränä on 2017-01-02. Toistossa edetään joka toinen alkio kerrallaan. Parillinen indeksi sisältää tarkastelujakson aloituspäivämärään ja pariton indeksi lopetuspäivämäärän. Muuttujaan *invalidAmountOfUnits* lasketaan läpikäytävän aikajakson puutteellisten myyntien määrä.

Tarkempi tietokantavastaus tallentuu kokoelmaan *dbResult* (ohjelmointiesimerkki 2 rivi 17). Tietokantavastauksesta käy ilmi puutteellisen rivin koodi sekä nimi, jotka voivat olla esimerkiksi lokaation koodi ja nimi. Tietokantavastauksen tulokset kirjoitetaan erillisille Excel-sivulle, jonka muodostamislogiikka ei ole tämän esimerkin kannalta oleellista. Metodissa *createSummaryPageValues* muodostetaan Excel-sivun yhteenvetosivulle tarvittava kuvaus (Map). *TransactionsValidator*-luokan avulla voidaan tehdä tarkastelua useasta eri näkökulmasta, kuten tuote- tai tuoteryhmätasolta. Tietokantakyselyn mallin (rivi 9) kohtaan "{:table}" voidaan asettaa mikä tahansa tietokannan taulu, jolloin myyntien puutteellisuutta tutkitaan kyseisestä näkökulmasta.

Ohjelmointiesimerkki 2.

```
1  public abstract class Validator {
2      public abstract void run();
3      public abstract Map<Integer, List<String>>
           createSummaryPageValues();
4      public abstract void createExcelReport(ValidatorExcelWriter
           validatorExcelWriter) throws IOException;
5  }
6
7  public class TransactionsValidator extends Validator{
8        public void run() {
9            final String CMD_EMPTY_SALES_ON_TIME_PERIOD =
                 "begin_trx.cube.slice(Fastorm::LocalDate.new('{:dateFrom}'),
```

```
            Fastorm::LocalDate.new('{:dateTo}')).slice(\"{:table}\",
            \"sum(transactions.sales_quantity) =
            null\").view(\"{:table}\").each{|t| puts \"#{t.code} /
            #{t.name}\"};";
10          TemplateUtil transactionsTemplate = new
                TemplateUtil(CMD_EMPTY_SALES_ON_TIME_PERIOD);
11          HashMap<String, String> options = new HashMap<>();
12          options.put("{:table}", this.tableName);
13          for (int i = 0; i < timePeriods.size(); i = i + 2) {
14            options.put("{:dateFrom}", timePeriods.get(i));
15            options.put("{:dateTo}", timePeriods.get(i + 1));
16            String transactionsQuery =
                  transactionsTemplate.replace(options);
17            HashSet<String> dbResult = new
                  HashSet<>(this.fastormQuery.runRunner(transactionsQuery));
18            this.invalidAmountOfUnits += dbResult.size();
19          }
20      }
21
22      public Map<Integer, List<String>> createSummaryPageValues(){
23        //Forms summary map using instance variables e.g.
              'invalidAmountOfUnits' and 'tableName'
24      }
25      public void createExcelReport(ValidatorExcelWriter
            validatorExcelWriter){}
26  }
27
28  public class ValidationMain {
29    public static void main(String[] args){
30        List<Validator> validators = new ArrayList();
31        validators.add(new TransactionsValidator("locations"));
32        validators.add(new ReferenceValidator("products"));
33        for(Validator validator : validators){
34          validator.run();
35        }
36      }
37  }
```

Case-yrityksen tuotteen käyttöliittymästä on mahdollista suorittaa Java-luokkia. *ValidationMain*-luokan suorittamisen jälkeen käyttöliittymään ilmestyy latausikkuna, mistä käyttäjä voi ladata excel-raportin. Excel-raportin kirjoittamiseen käytettiin Apache POI:n kirjastoja [29], joiden avulla on mahdollista ohjelmallisesti kirjoittaa erilaisia Microsoftin ohjelmistoihin soveltuvia dokumentteja, kuten Excel-raportteja. Excel-kirjoittimeen lisättiin erilaisia tapoja kirjoittaa tietorakenteita, kuten HashMappeja [27] Excel-tiedostoon.

		Summary	
Locations Sales Validation	**Number of locations**	**Status**	
Number of locations	6	N/A	
Locations without sales during 13 months	0	Valid	
Months with sales -%	100%	N/A	
Groups_1 Sales Validation	**Number of groups_1**	**Status**	
Number of groups_1	5	N/A	
Groups_1 without sales during 13 months	1	Immediate Attention Needed	
Months with sales -%	80%	N/A	
Groups_2 Sales Validation	**Number of groups_2**	**Status**	
Number of groups_2	63	N/A	
Groups_2 without sales during 13 months	11	Immediate Attention Needed	
Months with sales -%	85%	N/A	
Groups_3 Sales Validation	**Number of groups_3**	**Status**	
Number of groups_3	424	N/A	
Groups_3 without sales during 13 months	160	Immediate Attention Needed	
Months with sales -%	77%	N/A	
Negative Balances Validation	**Amount**	**Status**	
Rows containing negative balances from yesterdays file	4924	Attention Needed	
Rows containing negative balances last 30 days	120796	Attention Needed	
Balance transactions last 30 days	14129322	N/A	
Invalid Rows -%	0.9%	N/A	
Reference Validation	**Value**	**Status**	
Invalid Products Reference Codes	0	Valid	
Invalid Locations Reference Codes	0	Valid	
Delivery Validation	**Amount**	**Status**	
Open purchase orders more than 2 weeks late from yesterdays file	6998	Immediate Attention Needed	
Amount of null group_id fields in products table	**Amount**	**Status**	
Null row count	165364	Attention Needed	
Total rows count	247383	N/A	
Null rows / total rows -%	66.8%	Attention Needed	
Message Validation	**Period**	**Errors Found**	**Status**
Invalid Product_Location_Update reference codes	-3 days	73	Attention Needed
Redundant data validation (no transactions)	**Redundant Rows**	**Redundant Rows / All Rows -%**	**Status**
Groups_1	1	20.0%	Attention Needed
Product_locations	228213	35.7%	Attention Needed
Locations	0	0.0%	Valid
Products	91249	36.9%	Attention Needed

Kuva 13: Excel-raportin yhteenvetosivu. Raportti on muodostettu case-yrityksen asiakkaan datasta.

6.5 Työkalun muodostama raportti

Luvussa 5.3 on listattu asiantuntijahaastattelujen perusteella muodostetut validointitapaukset. Kyseiset tapaukset on ohjelmoitu kirjoittamaan Excel-raportin yhteenvetosivulle tiivistelmän löydöksistään. Excel-raportin yhteenvetosivu esitellään kuvassa 13. Kuvassa oleva Excel-raportti on validointityökalun muodostama raportti ja se on muodostettu case-yrityksen todellisen asiakkaan datasta.

Kaikki validointiluokat kirjoittavat tiivistelmän löydetyistä puutteista yhteenvetosivulle. Lisäksi osa luokista kirjaa tarkempia tietoja erillisille

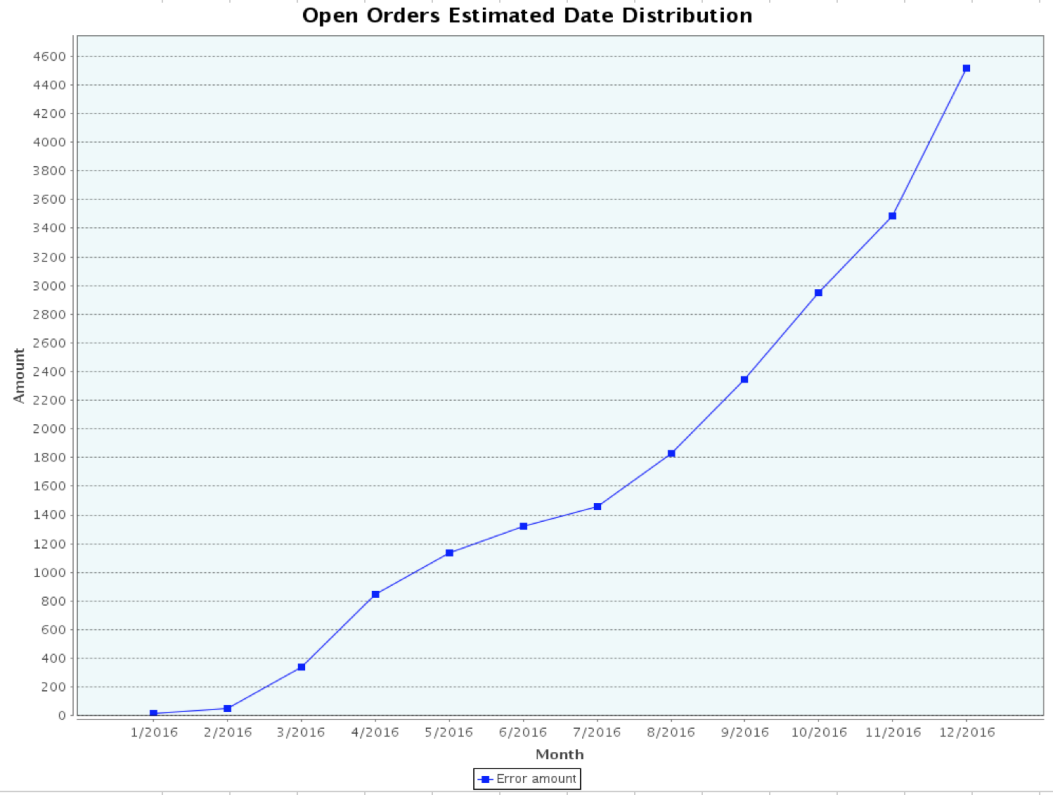
sivulle. Yhteenvetosivulle kirjataan jokaisen validointitapauksen yhteenveto erillisiin kokonaisuuksiin. Yhteenvetosivun tarkoitus on antaa nopea yleiskuva asiakkaan datan laadusta. Sivun avulla näkee kriittisimmät datavirheet, joita ovat myyntitapahtumiin liittyvät virheet. Ylimääräinen data (engl. redundant data) ei ole niin kriittistä kuin myyntitapahtumiin liittyvät virheet, sillä tällä datalla ei tehdä liiketoimintaan vaikuttavaa laskentaa. Ylimääräinen data voi mahdollisesti aiheuttaa tietovarastoa käyttävälle taholle hämmennystä ja hidastaa päätöksentekoa. Lisäksi ylimääräinen data vaikuttaa case-yrityksen tuotteen suorituskykyyn.

Kuvassa 13 olevan yhteenvetosivun neljä ensimmäistä tapausta liittyvät myyntitapahtumien validointiin. Validointitapausten lähdekoodi on edellisen sivun ohjelmointiesimerkissä. Jokainen tapauksista tarkastelee eri näkökulmista, puuttuuko myyntejä. Myyntejä tarkastellaan lokaatio-, sekä tuoteryhmätasoilla. Tuoteryhmiä on useissa eri tasoissa. Ensimmäinen tuoteryhmätaso voi tarkoittaa esimerkiksi virvoitusjuomia, toinen tuoteryhmätaso alkoholijuomia ja kolmas tuoteryhmätaso oluita. Jos myyntejä puuttuu jollain tasolla, se voi tarkoittaa useaa eri asiaa. Asiakkaan tiedonsiirtorajapinnassa voi olla virhe ja myyntidatan lähetyslogiikka on muodostettu väärin. Vaihtoehtoisesti virhe voi olla vastaanottavassa päässä ja tuoteryhmät ovat muodostettu väärin. Yksi vaihtoehto on myös, että asiakkaan tietokantaan on muodostettu ylimääräisiä lokaatioita tai tuoteryhmiä, joita ei käytetä liiketoiminnan suunnittelussa, mutta ovat jäänteitä mahdollisesta aiemmasta suunnittelutyöstä.

Negatiiviset saldot (Negative Balances Validation) ovat datavirhe tuotteiden saldoja kuvaavassa aineistossa. Saldot eivät saisi olla negatiivisia, koska se hyvin harvoin heijastaa mitään todellista liiketoiminnallista tilaa. Esimerkiksi olutta ei voi olla hyllyssä -20 kappaletta. Kuvan 13 kohdassa *Negative Balances Validation* nähdään huomattavan paljon negatiivisa saldoja. Negatiiviset saldot voivat johtua esimerkiksi siitä, että toimituksia ei olla kirjattu vastaanotetuiksi, jolloin saldotieto ei kuvaa todellista tilannetta. Vaihtoehtoisesti saldoja voidaan laskea manuaalisesti tarkastamalla hyllyjen ja varaston tilaa tietyin väliajoin, jolloin ne ovat ajan tasalla vain laskentapäivinä.

Viitteiden validointi (Reference Validation) tarkistaa tuote-, sekä lokaatiotaulujen viitesarakkeita. Sarakkeita käytetään kuvaamaan tosielämän tilannetta, missä jokin tuote tai lokaatio muistuttaa jotain toista tuotetta tai lokaatiota. Lokaatiot muistuttavat toisiaan esimerkiksi siinä tapauksessa, kun ne ovat saman ryhmän myymälöitä, ne ovat kooltaan lähellä toisiaan ja niiden sijainnit muistuttavat toisiaan. Viitteitä käytetään monesti esimerkiksi silloin, kun uusi tuote tai myymälä avataan, jolloin tuotteella tai myymälällä ei ole vielä myyntejä olemassa. Kun tuote tai myymälä viittaa toiseen tuotteeseen tai myymälään, viittaava tuote tai myymälä perii viitatun tuotteen tai myymälän myynnit, jolloin sille voidaan laskea automaattisesti ennusteita. Viitteet validoidaan tarkistamalla, ovatko kaikki viitearvot olemassa tuote- tai lokaatiotaulussa.

Date	1/2016	2/2016	3/2016	4/2016	5/2016	6/2016	7/2016	8/2016	9/2016	10/2016	11/2016	12/2016
Amount	16	47	341	848	1134	1320	1462	1827	2347	2947	3484	4519



Kuva 14: Avointen ostotilausten määrän kuvaus päivämäärän mukaan. Kuvauksessa y-akselilla myöhässä olevien avointen ostotilausten määrä ja x-akselilla aika kuukausittain.

Toimitusten validoinnissa (Delivery Validation, kuva 14) tarkistetaan, kuinka moni eilisistä avoimista ostotilauksista on myöhässä yli kaksi viikkoa. Avoimet ostotilaukset ovat tilauksia, jotka eivät ole vielä saapuneet päämääräänsä. Avoimilla ostotilauksilla on arvioitu toimituspäivämäärä, jonka avulla kyseinen validointitapaus muodostetaan. Case-yrityksen tuote ottaa huomioon avoimet ostotilaukset muodostaessaan automaattisia tilausehdotuksia vähentämällä tuotteelle lasketusta tilausehdotuksesta avointen ostotilausten määrän. Vanhojen avointen ostotilausten validointitapaus esittää ostotilausten määrän kuvauksen saapumispäivämäärän mukaan omalla sivullaan Excel-raportissa.

Jos tuotteella on myyntejä, mutta sillä ei ole tuoteryhmää, myyntien tarkastelu tuoteryhmätasolla näyttää vääriä myyntimääriä. Kuvan 13 kohta ”Amount of null group_id fields in products table” kertoo, kuinka monella tuotteella ei ole tuoteryhmää.

Tuotetta ja lokaatioita yhdistää tuote-lokaatio-taulu. Avainarvot ovat

tuote- sekä lokaatiokoodi, joka tarkoittaa, että jokaiselta tuote-lokaatiotaulun riviltä on viite tuotetauluun sekä lokaatiotauluun. Tuote-lokaatio taulusta on lisäksi viite toimittajatauluun. Viestivalidoinnissa (Message Validation) tutkitaan tuote-lokaatio dataa. Datasta etsitään tuote-, lokaatio-, sekä toimittajakoodeja, jotka eivät ole oikeita eli eivät ole olemassa niitä vastaavissa tauluissa. Kuvassa 13 oleva *Period -3 days* tarkoittaa, että tuote-lokaatio dataa on tutkittu viimeisen kolmen päivän ajalta.

Ylimääräisen datan (redundant data) validointitapauksissa etsitään eri perspektiiveistä turhaa dataa. Data määritellään turhaksi, mikäli sillä ei ole yhtään transaktioita koko historian aikana. Turhaa dataa etsitään tuoteryhmistä (Groups1), tuote-lokaatioista (product-locations), lokaatioista (locations), sekä tuotteista (products). Muut kuin ensimmäinen tuoteryhmä päätettiin jättää toistaiseksi ylimääräisen datan tarkastelusta pois, mutta voidaan tarvittaessa lisätä tarkasteluun myöhemmin.

7 Työkalun arviointi

Tässä luvussa arvioidaan työkalun ominaisuuksia vertaamalla sitä tutkimuksissa esille tuotuihin muihin datan laatupuutteita etsiviin työkaluihin. Työkalun toimintaa testattiin käytännössä case-yrityksen asiakashankkeissa haastattelemalla hankkeissa työskentelevää case-yrityksen henkilökuntaa. Haastatteluissa kysyttiin ensin, mitä datan laatuvirheitä validoinnin aikana ilmeni. Tämän jälkeen projektitiimiläisille esiteltiin, mitä laatuvirheitä työkalu löysi tietokannasta. Tällä tavalla löydettiin päällekkäisyyksiä työkalun ja projektitiimiläisten löydöksistä, jolloin työkalun toiminta osoitettiin toimivaksi. Lisäksi työkalu löysi datan laatuvirheitä, joita projektitiimiläiset eivät olleet itse löytäneet. Luvun lopussa käydään läpi työkalun saama palaute.

7.1 Työkalun teknisten ominaisuuksien vertailu aiempiin tutkimuksiin

Bareteiron ja Galhardasin kirjallisuuskatsauksessa [6] *yleisiksi* määritellyistä ominaisuuksista (ks. luku 4.2) kaikki paitsi metadatan säilöntä on toteutettu case-yrityksen ETL-prosessiin tai tuotteeseen. Seuraavissa kappaleissa käydään läpi, mitkä kyseisen tutkimuksen ominaisuuksista ovat osana itse työkalua.

Tutkijat mainitsevat, että työkalun käytettävyyden ja käyttöliittymän on oltava ymmärrettävää [6]. Tutkielmassa tehdyn työkalun käyttäminen tapahtuu case-yrityksen käyttöliittymästä. Käyttäminen tapahtuu valitsemalla ajo ja painamalla *Start*. Varsinainen raportti muodostetaan Exceliin. Toisen haastattelukierroksen aikana Excelin yhteenvetosivua kuvattiin *selkeäksi* ja *ytimekkääksi*.

Kirjallisuuskatsauksessa kerrotaan, että työkalussa on hyvä olla joukko valmiita validointisääntöjä, sekä tuki jollekin ohjelmointikielelle [6]. Työkalun

arkkitehtuuri mahdollistaa uusien validointisääntöjen lisäämisen helpohkosti, sillä validointitapaukset ovat toisistaan irrallisia, mutta pohjautuvat samaan yläluokkaan. Uusia validointitapauksia voi lisätä työkaluun ohjelmoimalla niitä Java-ohjelmointikielellä. Käytännössä jos käyttäjä haluaa lisätä työkaluun uusia validointitapauksia, hänellä on oltava hyvät teknilliset valmiudet.

Yhtenä ominaisuutena tuodaan esille suorituksen jäljitys ja lokitus [6]. Työkalun suoritusta on mahdollista seurata case-yrityksen käyttöliittymän kautta lokitiedostosta. Jokainen luokka kirjoittaa lokiin lähettämänsä tietokantakyselyt, jolloin suoritusta voidaan seurata. Työkaluun ohjelmoitiin poikkeusten hallinta pääasiallisesti *try-catch*-lausekkeen avulla. Kaikki poikkeukset kirjataan lokiin.

Yhtenä ominaisuutena mainitaan *suorituskyky ja skaalautuvuus* [6]. Työkalussa pyritään käyttämään paljon *Set-* tai *Map*-tietorakenteita. Tiedon hakeminen ja lisääminen näihin tietorakenteisiin on vakioaikaista hajautusfunktion ollessa hyvä. Työkalu on toiminut isoissa asiakasympäristöissä ja sen suorituksen nopeuttaminen ei ole ollut toistaiseksi aiheellista, sillä työkalu tekee validointitarkistukset merkittävästi ihmistä nopeammin. Muut luvussa 4.2 esitellyt ominaisuudet ovat case-yrityksessä enemmän ETL-prosessiin liittyviä ominaisuuksia.

Useiden jo olemassa olevien työkalujen tekninen toiminta pohjautuu SQL-kieleen, kuten esimerkiksi Arktos, Ajax ja FraQL [25]. Työkalujen toiminta perustuu tietokantakyselyihin SQL-kielellä. Tässä tutkielmassa tehdyn työkalun toimintaperiaate on tietokantakyselyt Java- ja Ruby-ohjelmointikielillä. Lisäksi tietokantakyselyissä käytetään case-yrityksessä tehtyä TQL-kyselykieltä.

Tutkielmassa tehdyssä työkalussa yhtenä ominaisuutena on puuttuvien myyntien havainnointi, joka on ominaisuutena esimerkiksi FraQL-työkalussa [25]. FraQL osaa automaattisesti täyttää puuttuvia arvoja tilastollisin keinoin. Lisäksi [16] mainitsee puuttuvien myyntien olevan datan laatuongelma.

Ylimääräisen datan sekä väärien viitteiden kerrotaan olevan laatuongelma artikkelissa [16], mitkä lisättiin validointitapauksiksi tutkielmassa toteutettuun työkaluun.

Vanhentuneet arvot ovat datan laatuongelma [26]. Vanhentuneita arvoja sisältävällä datalla tarkoitetaan arvoja, jotka eivät kuvaa tämänhetkistä oikean elämän tilannetta. Työkaluun lisättiin tarkistus, joka tutkii yli kaksi viikkoa myöhässä olevia toimituksia. Kyseiset tapaukset voivat tarkoittaa esimerkiksi tilannetta, missä henkilö on unohtanut kuitata toimituksen saapuneeksi. Tällöin toimitusta kuvaava rivi edelleen lähetetään case-yritykselle ja toimituksen oletetaan edelleen saapuvan määränpäähän. Saapuvat toimitukset huomioidaan uusia tilausehdotuksia generoitaessa.

Ajax-, FraQL- ja IntelliClean-työkalut havaitsevat duplikaatteja [25]. Case-yrityksen ETL-prosessi puhdistaa automaattisesti duplikaatteja masterdatasta avainarvojen avulla. Transaktiodatasta ei puhdisteta automaattisesti duplikaatteja, sillä jokaista transaktiota pidetään itsenäisenä tapahtuma-

na, eikä transaktioille yleensä muodosteta uniikkia avainta, minkä avulla duplikaatit voitaisiin karsia automaattisesti.

Työkaluun ei ohjelmoitu validointitapausta duplikaattitransaktioiden havaitsemiseen tai puhdistamiseen, mutta se voi olla hyödyllistä rakentaa tulevaisuudessa. Ominaisuus voi pyrkiä etsimään tietokannasta esimerkiksi samoja oikeaa elämää kuvaavia myyntitapahtumia, jotka kirjoitetaan Relex-ohjelmistoon useana myyntitapahtumana, ellei toisenlaista logiikkaa erikseen määritellä. Duplikaatteja on mahdollista tunnistaa yhteneväisyysfunktion avulla [47, 8]. Ideana yhteneväisyyden tarkastamisessa on tehdä merkkijonovertailua kahden tietokantarivin sarakkeiden välillä.

7.2 Mittaustulokset

Tässä tutkielmassa halutaan mitata, kuinka hyvin toteutettu työkalu löytää datavirheitä tai -puutteita jollekin asiakkaalle tehdystä tietovarastoasennuksesta. Mittaamisprosessia mietittäessä päädyttiin valitsemaan kahden eri lähestymistavan väliltä. Ensimmäisessä lähestymistavassa suunniteltiin datasarjaa, mihin piilotetaan datavirheitä. Nämä virheet pystytään laskemaan manuaalisesti ja kirjaamaan ylös. Tämän jälkeen datasarja voidaan kirjoittaa tietovarastoon, ajaa työkalu ja tutkia, kuinka suuren osan datavirheistä työkalu löytää. Tällä tavalla mitattaisiin työkalun onnistumista lavastetussa olosuhteessa.

Toisena lähestymistapana suunniteltiin työkalun käyttämistä projektivaiheessa olevissa asiakashankkeissa, joissa hanke on vasta alkanut. Kun työkalu on ajettu asiakasympäristössä, asiakashankkeen projektitiimiä voidaan haastatella. Haastattelun tavoitteena on saada selville, minkälaisia datavirheitä projektitiimiläiset ovat löytäneet tietovarastosta. Samalla selvitetään, millä keinoilla datan validointia tehdään hankkeessa. Tällöin on mahdollista löytää päällekkäisyyksiä työkalun sekä projektitiimiläisten löydöksistä, jolloin voidaan arvioida työkalun toimivuutta. Päällekkäisellä löydöksellä tarkoitetaan esimerkiksi tilannetta, jossa projektitiimiläinen on havainnut joltain lokaatiolta puuttuvan myyntiä ja työkalu on löytänyt saman puutteellisuuden.

Tutkielmassa päädyttiin jälkimmäiseen lähestymistapaan. Hyötynä tästä lähestymistavasta on valmis datasarja, joka kuvaa valmiiksi jotain tosielämän tilannetta ja liiketoimintamallia. Lähestymistavassa saadaan suoraan palautetta työkalun mahdollisilta tulevilta käyttäjiltä, mikä on arvokasta jatkokehityksen kannalta. Hyötynä lähestymistavassa on saada sana leviämään case-yrityksessä työkalun olemassaolosta ja sen mahdollisista hyödyistä. Lisäksi lähestymistapaan päädyttiin, sillä oikeaa liiketoimintaa kuvaavaa datasarjaa on työlästä muodostaa ja sen muodostaminen olisi vaatinut jonkun työntekijän perehdyttämistä ja monia tunteja aikaa. Työkalun toimintaa arvioitiin kolmessa asiakashankkeessa haastattelemalla hankkeiden työntekijöitä (liite 2). Haastattelut pidettiin case-yrityksen tiloissa ja ne kestivät noin tunnin ajan. Tutkielman tekijä kirjoitti haastatteluista muistiinpanot.

7.2.1 Ensimmäinen asiakashanke

Ensimmäisessä asiakashankkeessa työskenteli pääasiallisesti kaksi case-yrityksen työntekijää: projektipäällikkö sekä tekninen projektipäällikkö. Projektipäällikön vastuualueisiin kuuluu muun muassa liiketoiminnallisen logiikan muodostaminen asiakkaalle, sekä aikatauluissa pysyminen. Tekninen projektipäällikkö vastaa muun muassa ETL-prosessin muodostamisesta sekä käyttöliittymän muokkaamisesta asiakkaan tarpeiden mukaiseksi. Asiakashanke alkoi keväällä 2017.

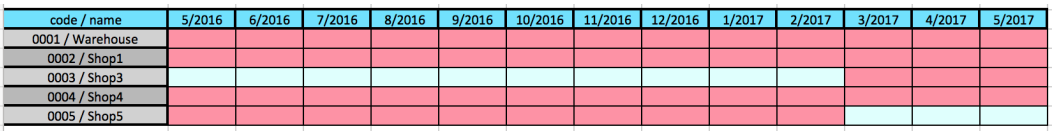
Tietovarastoon kirjoitetaan dataa päivittäin. Kysyttäessä kuinka dataa validoidaan hankkeessa projektipäällikkö kertoi heillä olevan käytössä erilaisia näkymiä, joiden avulla he voivat huomata esimerkiksi puutteita myynneissä. Projektipäällikkö on neuvonut asiakkaan työntekijöitä validoimaan masterdataa tarkistamalla, ovatko arvot oikein sekä näyttämällä, kuinka puuttuvia arvoja voidaan etsiä. Projektipäällikkö oli tehnyt masterdatan validointia asiakkaan kanssa muodostamalla case-yrityksen ohjelmiston käyttöliittymään näkymiä, joihin oli lisätty esimerkiksi tuotteeseen liittyviä ominaisuuksia. Näkymien avulla tietokannassa oleva data havainnollistetaan ihmiselle luettavampaan muotoon.

Projektitiimiläisiltä kysyttiin, minkälaisia datavirheitä heillä on ollut liittyen masterdataan. He kertoivat, että asiakkaalla oli aiemmin ollut ongelmia liittyen tuoteryhmiin. Jotkin tuotteet olivat väärissä tuoteryhmissä ja osalla tuotteista ei ollut tuoteryhmää. Lisäksi asiakas toi omasta ERP-järjestelmästään tietovarastoon tuotteita, jotka eivät olleet myytävänä. Kyseiset tuotteet eivät tuo asiakkaalle lisäarvoa, sillä asiakas käyttää tietovarastoa pääasiassa automaattiseen täydentämiseen eli tilasuehdotusten generointiin. Tilausehdotuksia muodostetaan tuotteille historiallisten myyntien ja niistä laskettujen ennusteiden perusteella, avointen ostotilausten sekä saldojen perusteella. Jos tuotteet eivät ole myynnissä, niille ei muodosteta tilausehdotuksia.

Seuraavaksi projektitiimiläiset kertoivat, mitä transaktiodataan liittyviä ongelmia asiakkaalla on ollut. Asiakaakkaalla oli aluksi teknisiä vaikeuksia lähettää avoimia ostotilauksia tietovarastoon luettavaksi. Toinen transaktioihin liittyvä ongelma oli tuotekoodien puuttuminen joistain myyntitapahtumista. Tuotekoodi on pakollinen kenttä, kun myyntitapahtumia siirretään case-yrityksen tietovarastoon. Jos tuotekoodi puuttuu datasta, rivi ohitetaan. Projektin tekninen projektipäällikkö kertoi heidän tehneen järjestelyn, missä kyseisistä riveistä ilmoitetaan asiakkaalle automaattisesti.

Kuvassa 15 on ensimmäisen asiakkaan kuukausitasoiset puuttuvat myynnit eri lokaatioissa. Kuvassa oleva data on obfuskoitu asiakkaan identiteetin suojaamiseksi. Kuva on otettu työkalun muodostamalta välilehdeltä nimeltään *Missing Sales locations*, joka muodostetaan havaittaessa myyntiaukkoja lokaatioissa. Excel-raporttiin kirjoitetaan ainoastaan lokaatioita, millä on ollut joitain puutteita viime vuoden aikana. Kuvassa näkyvä ensimmäinen

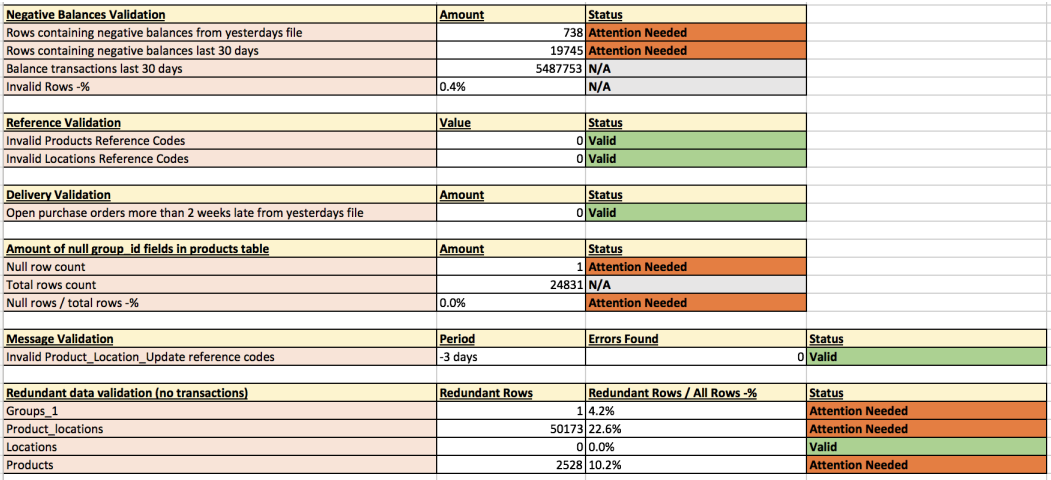
punainen rivi kuvaa puuttuvia myyntejä asiakkaan varastosta. Projektipäällikkö yllättyi tiedosta, ettei heillä ole myyntejä varastossa (kuvassa 15 ”0001 / Warehouse”). Muut puutteet olivat projektipäälliköllä tiedossa. Hän kertoi ettei lokaatioilla 0002 sekä 0004 tule olla myyntejä. Lokaatio 0003 suljettiin helmikuun lopussa, kun taas 0005 avattiin maaliskuussa.



code / name	5/2016	6/2016	7/2016	8/2016	9/2016	10/2016	11/2016	12/2016	1/2017	2/2017	3/2017	4/2017	5/2017
0001 / Warehouse													
0002 / Shop1													
0003 / Shop3													
0004 / Shop4													
0005 / Shop5													

Kuva 15: Validointityökalun löytämät myyntiaukot kuukausitasolla eri lokaatioissa. Data obfuskoitu asiakkaan identiteetin turvaamiseksi.

Kuvassa 16 on esitelty muita löydöksiä ensimmäisestä asiakashankkeesta. Kerrottaessa negatiivisista saldoista (kohta Negative Balances) projektipäällikkö kertoi havainneensa negatiivisia saldoja aiemmin järjestelmässä. Hän sanoi negatiivisten saldojen olevan ikävä laatuongelma, ja asiakkaan on hyvä korjata ongelma mahdollisimman nopeasti. Yhtä tuotetta ei oltu kiinnitetty mihinkään tuoteryhmään (kuvassa "Amount of null group_id fields in products table"). Projektipäällikkö pyysi antamaan kyseisen tuotteen tuotekoodin haastattelun jälkeen. Järjestelmästä löytyi 2529 ylimääräistä tuotetta, mistä projektitiimiläiset puhuivat haastattelun alkuvaiheessa. Järjestelmästä löytyi myös noin 50000 ylimääräistä tuote-lokaatioita.



Negative Balances Validation	Amount	Status
Rows containing negative balances from yesterdays file	738	Attention Needed
Rows containing negative balances last 30 days	19745	Attention Needed
Balance transactions last 30 days	5487753	N/A
Invalid Rows -%	0.4%	N/A

Reference Validation	Value	Status
Invalid Products Reference Codes	0	Valid
Invalid Locations Reference Codes	0	Valid

Delivery Validation	Amount	Status
Open purchase orders more than 2 weeks late from yesterdays file	0	Valid

Amount of null group_id fields in products table	Amount	Status
Null row count	1	Attention Needed
Total rows count	24831	N/A
Null rows / total rows -%	0.0%	Attention Needed

Message Validation	Period	Errors Found	Status
Invalid Product_Location_Update reference codes	-3 days	0	Valid

Redundant data validation (no transactions)	Redundant Rows	Redundant Rows / All Rows -%	Status
Groups_1	1	4.2%	Attention Needed
Product_locations	50173	22.6%	Attention Needed
Locations	0	0.0%	Valid
Products	2528	10.2%	Attention Needed

Kuva 16: Validointityökalun löytämät muut tulokset. Kuvakaappaus otettu yhteenvetosivulta.

7.2.2 Toinen asiakashanke

Toisessa asiakashankkeessa työskenteli tekninen projektipäällikön ja projektipäällikön lisäksi vanhempi projektipäällikkö. Vanhemman projektipäällikön rooli hankkeessa oli seurata ylätasolta, että hanke pysyy aikataulussa, sekä auttaa ja tukea muita projektitiimiläisiä. Molemmat pääasialliset työntekijät olivat uusia tekijöitä, jonka takia vanhemman projektipäällikön mukana oleminen oli perusteltua. Lisäksi hankkeessa oli ollut mukana auttamassa vanhempia teknisiä projektipäälliköitä, jotka pystyivät antamaan teknistä tukea. Hanke alkoi keväällä 2017. Asiakkaan pääasiallinen käyttötarkoitus Relex-ohjelmistolle on tuotannon suunnittelu. Käytännössä tämä tarkoittaa sitä, että Relexin laskemien ennusteen perusteella asiakas tietää, kuinka paljon tuotteita on tuotettava minäkin päivänä. Asiakkaita, joiden pääasiallinen käyttötarkoitus Relex-ohjelmistolle on tuotannon suunnittelu, kutsutaan case-yrityksessä ennusteasiakkaiksi.

Haastattelu aloitetiin kysymällä, miten projektitiimiläiset ovat validoineet dataa. Tekninen projektipäällikkö kertoi validoineensa dataa ETL-pakettiin ohjelmoitujen yksikkötestien avulla. Hän kertoi seuranneensa ETL-prosessista muodustavaa lokia, johon kirjataan automaattisesti tietoa, kuinka tiedostojen prosessointi etenee, sekä virheilmoitukset. Tekninen projektipäällikkö kertoi tehneensä uskottavuustarkistuksia. Uskottavuustarkistuksilla pyritään selvittämään tiedon paikkansapitävyys. Lisäksi tekninen projektipäällikkö ja projektipäällikkö kertoivat validoineensa dataa näkymien avulla sisäisesti, sekä yhdessä asiakkaan kanssa. Näkymiltä pyrittiin katsomaan, puuttuuko joitain arvoja. Asiakas vertasi Relexissä näkyviä arvoja omaan järjestelmäänsä.

Seuraavaksi projektitiimiläisiltä kysyttiin, minkälaista masterdataa heillä on käytössä. Tiimiläiset kertoivat vastaanottaneensa tuote-, tuotehierarkia-, asiakas-, toimittaja-, sekä tuote-asiakas-dataa. Ennusteasiakkaille tyypillisesti tehdään asiakkaiden ryhmittelyä. Näistä asiakasryhmistä muodostetaan Relex-ohjelmistoon lokaatiot. Ryhmittelyn tarkoituksena on parantaa ennustetarkkuutta, sillä usein ennusteasiakkailla on tuhansia asiakkaita. Yhdelle asiakkaalle myydään harvoin, jolloin ennustamisesta tulisi tuote-asiakastasolla epätarkkaa. Tekninen projektipäällikkö kertoi asiakasryhmittelyn olevan haastava toteuttaa, sillä ryhmittelyn suunnitelma oli puutteellinen ja ryhmittelylogiikka määriteltiin uudelleen projektin aikana. Lisäksi tiimiläiset kertoivat tuotehierarkian muuttuneen projektin aikana asiakkaan toimesta. He kertoivat masterdatan olevan pääosin hyvälaatuista.

Transaktiodatan osalta projektitiimiläiset kertoivat asiakkaan lähettävän saldoja, myyntejä, avoimia ostotilauksia sekä avoimia myyntitilauksia. He kertoivat transaktioiden integraation olleen haastava prosessi. Asiakas halusi Relexiin paljon erilaisia myyntitapahtumiin liittyvää lisätietoa, joka ei ole Relexin toiminnan kannalta merkityksellistä. Datan sisällössä oli ongelmia. Tiimiläiset kertoivat saldoissa olleen virheellisiä arvoja, sekä myyn-

titapahtumien arvoissa olleen sekaisin netto- ja bruttoarvoja. He kertoivat vastaanottaneensa projektin alussa vain avoimia myyntitilauksia, joihin on tehty jokin muutos aiempaan nähden, mutta Relex-ohjelmistoon halutaan kirjoittaa kaikki päivittäiset avoimet myyntitilaukset. Lisäksi he mainitsivat myyntitapahtumien arvojen olleen epätarkkoja valuuttamuunnosten takia. Tekninen projektipäällikkö mainitsi projektin aikana Relexiin ilmaantuneen duplikaattitransaktioita. Hän kertoi sen olleen integraatio-ongelma, joka johtui hänen tietämättömyydestään.

Tiimiläiset kertoivat, että tuotehierarkian ja asiakasryhmittelyn lisäksi projektin aikana iteroitiin paljon ketju-, sekä maa-tasoja. Ketju koostuu yhdestä tai useammasta lokaatiosta. Maa koostuu yhdestä tai useammasta ketjusta. Asiakkaan järjestelmässä logiikka oli erilainen, minkä takia integraatio ei ollut suoraviivaista.

Seuraavaksi haastattelussa siirryttiin tutkimaan validointityökalun tuottamia tuloksia. Asiakkaalla oli työkalun ajo hetkellä käytössä 52 lokaatiota. Työkalu tutki kuukausittaisia myyntejä lokaatiotasolla ja toi esille excel-raporttiin mahdollisia puutteita myyntidatassa. 46 lokaatiolla oli yksi tai useampi kuukauden kokoinen myyntiaukko viimeisen vuoden aikana. Projektipäällikkö kertoi suuren osan asiakkaan tuotteista olevan kausituotteita, jolloin tuotteella ei välttämättä ole yhtään myyntiä kesäisin, mutta talvisin sille on paljon kysyntää. Tämä selittää työkalun löytämät myyntiaukot, eikä kyseessä ollut datan laatuvirhe.

Työkalu ei löytänyt virheitä saldoista tai viitteistä. Työkalu toi esille avoimia ostotilauksia, jotka ovat yli kaksi viikkoa myöhässä (kuva 17). Avoimia ostotilauksia käytetään laskemaan automaattisia tilausehdotuksia, mutta kyseinen asiakas ei käytä tilausehdotuksia. Projektipäällikkö kertoi vanhentuneiden ostotilausten olevan tiedossa oleva ongelma. Ostotilaukset tuodaan Relexiin vain raportointitarkoituksia varten. Ostotilausten perusteella Relex laskee saldon projektion, joka tarkoittaa oletettua saldoa jollekin päivälle, jos ostotilaus on vastaanotettu ajallaan. Vanhentuneet ostotilaukset voivat vääristää saldon projektiota.

Kuvassa 17 nähdään asiakkaalla olevan 1054 tuotetta, joista 36:lla puuttui kiinnitys tuoteryhmään työkalun suoritushetkenä. Sekä projektipäällikkö että tekninen projektipäällikkö olivat yllättyneitä tästä havainnosta. Havainto selittää osittain tuoteryhmätasoilla löydettyjä puuttuneita myyntejä. Projektitiimiläiset pyysivät listauksen tuotekoodeista, joilta puuttui tuoteryhmäkiinnitys.

Työkalu löysi dataa, jolla ei ollut yhtään transaktiota, minkä työkalu ilmoitti olevan mahdollisesti turhaa dataa. Tutkielman tekijä mainitsi kyseisen datan olevan Relexissä todennäköisesti hyödytöntä ja sen aiheuttavan mahdollisesti suorituskykyongelmia muun muassa näkymien latausajoissa. Projektipäällikkö sanoi asiakkaan haluavan pitää datan Relexissä, sillä asiakas tekee Relexissä tuotannonsuunnittelua. Kaikilla tuotteilla tai lokaatioilla ei tällä hetkellä ole myyntiä, mutta sitä voi tulla myöhemmin.

Delivery Validation	Amount	Status
Open purchase orders more than 2 weeks late from yesterdays file	11	Immediate Attention Needed

Amount of null group_id fields in products table	Amount	Status
Null row count	36	Attention Needed
Total rows count	1054	N/A
Null rows / total rows -%	3.4%	Attention Needed

Message Validation	Period	Errors Found	Status
Invalid Product_Location_Update reference codes	-3 days	0	Valid

Redundant data validation (no transactions)	Redundant Rows	Redundant Rows / All Rows -%	Status
Groups_1	1	25.0%	Attention Needed
Product_locations	50865	92.8%	Attention Needed
Locations	8	15.4%	Attention Needed
Products	364	34.5%	Attention Needed

Kuva 17: Toisen asiakashankkeen tuloksia. Tuloksista näkyy myöhässä olevia avoimia ostotilauksia, tuotteita, joilta puuttuu kiinnitys tuoteryhmään, sekä ylimääräistä dataa. Kuvakaappaus rajattu työkalun muodostaman Excel-raportin yhteenvetosivulta.

7.2.3 Kolmas asiakashanke

Kolmannessa asiakashankkeessa työskenteli pääasiassa projektipäällikkö ja tekninen projektipäällikkö. Asiakashankkeen alussa oli mukana vanhempi tekninen projektipäällikkö, joka auttoi muun muassa määrittelemään tarvittavat tiedonsiirtorajapinnat. Hanke alkoi joulukuussa 2016. Asiakkaan pääkäyttötarkoitus Relex-ohjelmistolle on muodostaa automaattisia tilausehdotuksia. Asiakkaan lisäksi hankkeessa oli mukana asiakkaan palkkaama IT-talo, jolle oli ulkoistettu projektin vaatimien tiedonsiirtorajapintojen muodostaminen case-yritykseen asiakkaan järjestelmästä.

Haastattelun alussa kysyttiin, onko dataa aloitettu validoimaan ja millä tavoin sitä validoidaan. Tiimiläiset kertoivat aloittaneensa validoinnin kesän 2017 loppupuolella. Validointia ei pystytty aloittamaan aiemmin, sillä asiakkaalta ei saatu riittävän kattavasti dataa. Tiimiläiset sanoivat ettei asiakkaan palkkaamalla IT-talolla ollut oikeuksia tehdä muutoksia asiakkaan järjestelmään, mikä hidasti tiedonsiirtoyhteyden rakentamista. Tekninen projektipäällikkö kertoi validoivansa dataa pääasiassa tutkimalla lokitiedostoja. Hän kertoi validoinnin olevan haastavaa, sillä vastaanotettuja tiedostoja tulee satoja päivässä.

Projektipäällikkö kertoi validoineensa master- sekä transaktiodataa erilaisten näkymien avulla. Näkymiä on lähetetty asiakkaalle validoitavaksi. Asiakas validoi näkymiä vertaamalla Relexissä näkyviä arvoja heidän oman järjestelmänsä arvoihin. Projektipäällikkö kertoi kopioineensa validointinäkymät toisen asiakkaan ympäristöstä. Hän sanoi pitäneensä koulutuksen asiakkaalle, kuinka heidän tulisi tarkastella ja vertailla dataa Relexin ja asiakkaan järjestelmän välillä.

Masterdatan osalta projektitiimiläiset kertoivat vastaanottavansa tuote-,

tuote-lokaatio-, tuotehierarkia-, sekä toimittajadataa. Tekninen projektipäällikkö kertoi vastaanotetun datan olevan monimutkaisinta, mitä hänellä on missään aiemmassa projektissa ollut. Kaikki tiedostot sisälsivät SAP-terminologiaa, jota oli haastavaa ymmärtää ilman aiempaa kokemusta SAP-järjestelmistä.

Projektitiimiläisiltä kysyttiin, minkälaisia master-dataan liittyviä ongelmia hankkeessa oltiin havaittu. Tiimiläiset kertoivat tuotteiden myyntihintojen, toimitusaikojen ja toimittajien olleen projektin aikana väärin. Lisäksi projektin aikana vastaanotettiin duplikaattimasterdatarivejä, kuten tuote-lokaatio-dataa.

Projektitiimiläiset kertoivat vastaanottavansa transaktioiden osalta muun muassa saldoja, myyntejä, avoimia osto- ja myyntitilauksia. He kertoivat vastaanottaneensa projektin aikana duplikaattidataa saldojen osalta, sillä he olivat vastaanottaneet asiakkaan testi- ja tuotantoympäristöstä saldoja. Lisäksi tiimiläiset kertoivat vastaanottaneensa duplikaattirivejä myös myyntien osalta. Lisäksi myyntidatassa oltiin havaittu myyntiaukkoja.

Projektitiimiläiset kertoivat Relexin automaattisen ennuste- ja tilausehdotuslaskennan olevan käyttökelvotonta transaktioista johtuneiden ongelmien takia. Projektipäällikkö kertoi, että projekti ei olisi onnistunut, jos asiakas ei olisi saanut korjattua ongelmia. He kertoivat datan olevan haastattelun ajanhetkenä (lokakuu 2017) huomattavasti paremmalla mallilla kuin projektin alussa.

Validointityökalu löysi ympäristöstä myyntiaukkoja, vanhentuneita ostotilauksia sekä ylimääräistä dataa. Lisäksi yhdellä tuotteella ei ollut tuoteryhmäkiinnitystä. Projektitiimiläiset eivät olleet yllättyneitä yhdestäkään työkalun havainnosta. He olivat havainneet työkalun löytämät puutteet validoinnin aikana. Projektipäällikkö kysyi, onko kaikki vanhentuneet ostotilaukset mahdollista toimittaa hänelle haastattelun jälkeen. Tutkielman tekijä lupasi auttaa häntä löytämään kyseiset ostotilaukset.

7.3 Palaute

Ensimmäisessä asiakashankkeessa projektitiimiläiset kertoivat työkalun antavan hyvää palautetta järjestelmässä olevasta datasta. Projektipäällikkö kertoi jakavansa työkalun tulokset asiakkaalle. Sekä tekninen projektipäällikkö että projektipäällikkö kertoi raportin edistävän hankkeessa käytettävän datan laatua. Tekninen projektipäällikkö kertoivat lisäksi yhteenvetosivun olevan hyvä ja antavan ytimekkään kuvan järjestelmän datapuutteista.

Toisen asiakashankkeen projektitiimiläiset sanoivat yhteenvetosivun olevan selkeä. He kertoivat oppineensa työkalun yhteenvetosivun avulla, mitä tapauksia on hyvä validoida. Heidän mielestään case-yritykseen olisi hyvä saada lisää ohjeita ja käytänteitä, miten datan validointia tulisi tehdä ja mihin asioihin olisi hyvä kiinnittää huomioita. He haluaisivat Relexiin tehtävän automaattisesti datan validointinäkymät, joiden avulla dataa voi tutkia

sisäisesti, sekä yhdessä asiakkaan kanssa.

Kolmannen asiakashankkeen projektitiimiläiset sanoivat työkalun antavan hyvän yleiskuvan dataongelmista. He kertoivat yhteenvetosivulta voivan nähdä suoraan, mitä dataongelmia tulisi tutkia tarkemmin. Projektitiimiläiset sanoivat lisäksi, että työkalu olisi hyvä suorittaa validoinnin alussa. Haastattelun aikaan validointi oli ollut jo käynnissä pidemmän aikaa, jolloin työkalun tekemät löydökset oltiin jo havaittu.

8 Johtopäätökset

Tässä luvussa tehdään yhteenveto tutkielman aikana tehdyistä tärkeimmistä löydöksistä. Lisäksi luvussa muodostetaan jatkokehitysehdotukset työkaluun, sekä case-yrityksen validointiprosesseihin liittyen.

8.1 Tärkeimmät löydökset

Tekniset projektipäälliköt kertoivat haastatteluissa, kuinka he validoivat dataa. Kaikki haastatteluihin osallistuneet tekniset projektipäälliköt kertoivat tarkastelleensa case-yrityksen tuotteen lokeista, muodostuuko datan muunnos- tai kirjoitusvaiheessa virheitä lokeihin. Luvussa 3.3.1 kerrottiin tutkimuksen [6] perusteella, minkälaiset datavirheet on mahdollista karsia hyvän tietokantaskeeman suunnittelun seurauksena. Case-yrityksen lokeihin kirjaantuu virhe useassa luvussa 3.3.1 esitellyistä tapauksista. Jos datariviltä puuttuu avainarvo tai datarivin sarake sisältää eri datatyypin kuin tietokantaan määritellyn, rivi kirjataan lokiin virheellisenä. Lisäksi datarivi voidaan ohittaa, jos datarivin sarakkeessa oleva arvo ei kuulu tietokannan skeeman mukaiseen arvojoukkoon.

Taulukossa 4 tiivistetään datan validoinnin löydökset kolmessa eri asiakashankkeessa. Puuttuvia myyntejä löytyi jokaisesta tietovarastoista. Asiakashankkeessa 1 varastolokaation puuttuvat myynnit yllättivät projektitiimiläiset. Kaikista tietovarastoista löytyi tuotteita ilman tuoteryhmää, joka tuli yllätyksenä toisen hankkeen työntekijöille. Jokaisesta hankkeesta löydettiin ylimääräistä dataa, joka oli tiedossa hankkeissa työskennellyille. Hankkeiden 2 ja 3 tietovarastoista löytyi myöhässä olleita ostotilauksia, joka ei yllättänyt henkilökuntaa.

Toisessa ja kolmannessa hankkeessa datan validoinnin aikana oltiin löydetty virheitä, mitä työkalu ei löydä. Toisessa ja kolmannessa hankkeessa oltiin löydetty duplikaattitransaktioita. Toisessa hankkeessa työskennelleet kertoivat netto- ja bruttoarvojen olleen sekaisin hankkeen aikana. Kolmannessa hankkeessa kerrottiin myyntihintojen ja toimitusaikojen olleen väärin.

Taulukko 4: Työkalun ja projektitiimiläisten datan validoinnin löydökset. Taulukossa symboli t tarkoittaa työkalun löydöstä, p projektitiimiläisten löydöstä ja y työkalun löydöstä, joka oli yllätys projektitiimille.

Validointitapaus	**1. Hanke**	**2. Hanke**	**3. Hanke**
Puuttuvat myynnit	t,p,y	t,p	t,p
Negatiiviset saldot	t,p		
Virheelliset viitteet			
Myöhässä olevat ostotilaukset		t,p	t,p
Tuoteryhmän puuttuminen	t,p	t,y	t,p
Tuote-lokaatiodatan koodit			
Ylimääräinen data	t,p	t,p	t,p

8.2 Validoinnin kehitysehdotukset

Kuvassa 18 esitellään datan validointiin liittyviä kehitysehdotuksia case-yrityksessä. Kuvassa vasemmalla on ETL-prosessiin liittyvät kehitysehdotukset. Tutkielman aikana havaittiin, ettei ETL-prosessi kirjoita lokitiedostoihin tietoa, mistä syystä dataa on puhdistettu. Prosessin aikana kerrotaan vain ohitettujen rivien määrä. Lisäksi ETL-prosessi päästää läpi liiketoiminnallisesti haitallista dataa, kuten vanhentuneita avoimia ostotilauksia, jotka otetaan huomioon uusia tilausehdotuksia muodostettaessa.

Kuva 18: Kehitysehdotukset jaoteltu kolmeen osaan: ETL-prosessiin, validointiin liittyviin käytänteisiin ja tutkielmassa toteutetun validointityökalun jatkokehitysideoihin.

Kuvan 18 keskellä kerrotaan validointikäytänteisiin liittyviä kehitysideoita. Haastattelujen aikana tutkielman tekijä sai selville, ettei case-yrityksessä olla muodostettu standardoituja validointinäkymiä, joiden avulla tietovarastohankkeiden työntekijät ja asiakkaat tarkastelevat datan validiteettia. Lisäksi datan validointia voi kouluttaa yrityksen työntekijöille sekä kirjoittaa aiheesta ohjeita sekä parhaita käytänteitä.

Kuvassa 18 oikeassa reunassa kerrotaan tutkielmassa toteutetun työka-

lun kehitysehdotuksia. Tutkielman luvussa 5.3 mainittiin validointityökalun yhdeksi ominaisuudeksi automaattisten validointinäkymien muodostaminen, joka jäi tämän tutkielman ulkopuolelle. Relex-ohjelmiston tietokantatauluista on mahdollista löytää asiakaskohtaiset sarakkeet, joiden sisältämän datan laatua voidaan tutkia. Työkaluun on mahdollista muodostaa validointitapaus, joka etsii asiakaskohtaiset sarakkeet ja muodostaa näiden perusteella validointinäkymiä. Ominaisuutta ei ohjelmoitu työkaluun, sillä case-yrityksessä ei ole vielä standardoituja validointinäkymiä, mitä työkalu voisi mukailla.

Tutkielmassa toteutettuun työkaluun ei ohjelmoitu duplikaattidatan havainnointiin liittyvää ominaisuutta, vaikka duplikaattitransaktiot ovat tietovarastohankkeissa usein ilmenevä ongelma. Validointityökalu ei tee muutoksia tietokantaan, vaan se havannoi ja raportoi mahdollisista dataongelmista. Työkalun käytön riskit pystyttiin tällä tavalla pitämään alhaisina. Jos duplikaattidataa havaitaan, se voidaan samalla puhdistaa ETL-prosessin aikana. Tutkielman tekijä ehdottaa ominaisuutta tuotekehitystiimille.

Tutkielmassa kehitetyn työkalun päätarkoituksena on löytää datavirheitä asiakkaan lähettämästä aineistosta heti tietovarastohankkeiden alkuvaiheessa. Luonnollisena jatkona on kehittää työkaluun ominaisuuksia, joiden avulla voidaan ylläpitää datan laatua. Case-yrityksessä käytetään hälytysjärjestelmää, joka seuraa asiakkaalta päivittäin vastaanotettuja tiedostoja. Hälytysjärjestelmä ei ota kantaa datan sisältöön, vaan sillä voidaan tarkistaa tiedostojen olemassaoloa ja niiden kokoja. Tutkielmassa muodostettua työkalua voidaan jatkokehittää päivittäisen datan sisällön valvontatyökaluksi. Päivittäisiä löydöksiä olisi mahdollista pitää tallessa tietokannassa, jonka avulla voidaan seurata, onko datan laatu heikentynyt vai parantunut ajan kuluessa.

8.3 Tutkielman luotettavuuden arviointi ja rajoitteet

Tutkielman teoreettisessa osiossa etsittiin ja analysoitiin useita aihealueeseen liittyviä lähteitä, jolloin tutkielman tekijälle muodostui ymmärrys datan validiteetista tietovarastoissa. Aihealueeseen liittyvä aiempi tutkimus auttoi tutkielman tekijää toteuttamaan validointityökalun case-yritykselle. Tutkielma pyrittiin jäsentelemään ja kirjoittamaan objektiivisesti tiedeyhteisön vaatimalla tarkkuudella. Tutkielman luvussa 5.1 esitettiin tapaustutkimuksen liityvät viisi vaihetta [33], jotka käytiin läpi myös tässä tutkimuksessa.

Koska tapaustutkimuksissa tehdään analyyttistä tutkimusta, ei tutkielmassa tehtyjä löydöksiä voida suoraan yleistää muihin tapauksiin. Tämä pätee tutkielmassa kehitettyyn työkaluun: sen toiminta rajoittuu case-yrityksen ohjelmiston tietokannassa olevan datan tulkitsemiseen. Tutkielmassa kehitetyn työkalun toimintaperiaate on mahdollista yleistää toimimaan mille tahansa tietokannalle, sillä sen toimintaperiaate perustuu tietokantakyselyihin, jotka muodostetaan mallien avulla. Tutkielman toimeksiantaja pyrkii ratkaisemaan työkalun avulla todellista ja relevanttia ongelmaa, minkä takia työkalua ei ollut järkevää kehittää yleismaailmalliseen muotoon. Toimek-

siantaja on voittoa tavoitteleva yritys, minkä takia ongelman esittäminen yleismaailmallisessa muodossa voisi vähentää toimeksiantajan kilpailukykyä.

Tämän tutkielman yhtenä rajoitteena voidaan pitää haastattelujen vähäistä lukumäärää. Haastatteluihin, minkä perusteella vaatimusmäärittely muodostettiin, osallistui vain seitsemän henkilöä. On mahdollista, että uusien haastattelujen myötä oltaisiin löydetty uusia validointitapauksia. Lisäksi työkalua testattiin vain kolmessa asiakashankkeessa, joka on tieteellisestä näkökulmasta pieni otoskoko.

Asiakashankkeissa käytetyn haastattelupohjan (liite 2) kysymys 8 osoittautui johdattelevaksi. Tutkielman tekijä pyrki kysymään palautetta validointityökalusta, mutta sen sijaan kysyi, onko työkalu *hyödyllinen* (engl. Is validation tool useful)? Kysymys johdattelee vastaajaa kertomaan työkalun olevan hyödyllinen ja mahdollinen kritiikki voi jäädä uupumaan. Parempi kysymys olisi ollut esimerkiksi "voisitteko antaa palautetta työkalusta?", jolloin hankkeiden henkilökunta voi vapaammin miettiä työkalun toimivuutta heidän tarpeisiinsa.

Teoreettisen ja empiirisen osion välillä on yhteneväisyyksiä. Case-yrityksessä ilmenneet tietovaraston datan laatuongelmat, kuten vanhentuneet arvot tai myyntiaukot, nousevat esille aiemmissa tutkimuksissa. Case-yrityksen datan laatuongelmia tietovarastohankkeissa voidaan pitää vertailukelpoisena aiempiin tutkimuksiin [26, 16].

Tutkielmassa hyödynnettiin kahta erityyppistä triangulaatiota: aineistotriangulaatiota kutsumalla haastatteluihin eri rooleista tulevia asiakashankkeissa työskenteleviä ihmisiä, ja menetelmätriangulaatiota keräämällä aineistoa asiantuntijahaastatteluista sekä aiemmista tutkimuksista. Tutkielman luotettavuutta olisi voinut lisätä käyttämällä tutkijatriangulaatiota ja teoriatriangulaatiota.

Tutkielmassa muodostetun työkalun hyötyjä case-yritykselle on haastavaa mitata. Asiakasprojektien henkilökuntaa haastateltaessa työkalun todettiin löytävän samoja virheitä kuin henkiökunta oli löytänyt. Työkalu löysi lisäksi joitain henkilökuntaa yllättäviä virheitä. Tämän perusteella työkalun voidaan sanoa tuottavan lisäarvoa yrityksen työntekijöille, mutta on vaikeaa mitata, kuinka paljon. Työkalun lisäarvoa voisi mitata esimerkiksi ajallisesti muodostamalla kaksi vertailuryhmää, joista toinen käyttäisi tutkielmassa toteutettua työkalua ja toinen ei. Kyseisessä tapauksessa vertailuryhmissä tulisi työskennellä kokemukseltaan samantasoista henkilökuntaa, ja asiakkaiden lähettämien aineistojen tulisi sisältää saman verran virheitä, jolloin vertailun voitaisiin sanoa olevan objektiivista. Tällaista ryhmittelyä on miltei mahdotonta tehdä käytännössä.

Työkalun hyötyä voidaan arvioida kysymällä hankkeiden validointivaiheen jälkeen jokaisessa projektissa, käytettiinkö validointityökalua ja kuinka kauan validointiin käytettiin aikaa. Vastaukset ryhmitellään erikseen projekteihin, missä työkalua käytettiin ja projekteihin, missä sitä ei käytetty. Tämän jälkeen mitataan validointiin käytettyä kokonaisaikaa kummassakin

ryhmittelyssä. Jos aineistoa kerätään monista projekteista, voidaan arvioida, nopeuttaako työkalun käyttö validointivaihetta vai ei.

8.4 Yhteenveto

Tutkimuskysymyksessä yksi kysyttiin, mitä haittoja huonolaatuisella datalla on tietovarastohankkeissa? Luvussa 3.1 kerrottiin huonolaatuisen datan liiketoiminnallisia seurauksia, joita ovat muun muassa ylimääräiset kulut, asiakkaiden tyytymättömyys ja järjestelmän käyttöönoton viiväsyminen [9]. Case-yrityksen ohjelmiston toiminta perustuu asiakkaiden lähettämään dataan, jolloin ohjelmiston tuottaman laskennan laatu ja sitä kautta ohjelmiston asiakkaalle tuovan lisäarvon laatu on parhaassa tapauksessa yhtä hyvää kuin vastaanotetun datan laatu. Jos datan laatua parannetaan asiakashankkeiden aikana, ohjelmisto tuottaa asiakkaille enemmän lisäarvoa.

Tutkielman toinen tutkimuskysymys oli, kuinka tietovarastohankkeiden osapuolet tarkistavat datan validiteettia? Luvussa 4.1 käytiin läpi kirjallisuudesta löytyneitä validointiprosesseja. Case-yrityksen asiakashankkeissa työskentelevää henkilökuntaa haastatellessa huomattiin puutteita case-yrityksen validointikäytänteissä. Case-yrityksen henkilökunnalta löytyy paljon tietoa, kuinka validointia tehdään, mutta kirjallisia ohjeita tai standardoituja validointinäkymiä ei ole olemassa. Varsinkin uusien työntekijöiden voi olla vaikeaa ymmärtää validoinnin tärkeyttä ja kuinka dataa validoidaan.

Tutkimuskysymyksessä kolme kysyttiin, onko mahdollista muodostaa automaatiota, joka ilmoittaa automaattisesti datavirheistä? Tutkielmassa muodostettu työkalu toimii case-yritykselle lähtökohtana datavirheiden löytämiseen asiakkaiden lähettämästä aineistosta. Asiakashankkeiden tyypilliset datan laatuvirheet kiteytettiin asiantuntijahaastatteluiden avulla luvussa 5.3. Tutkielmassa toteutettu työkalu löytää kyseiset virheet automaattisesti tietokannasta. Työkalun on mahdollista tukea validointiprosessia asiakashankkeissa ja se voi nopeuttaa virheiden löytämistä.

Tietovarastojen validiteetti on kiistatta ajankohtainen ongelma niin tiedemaailmassa kuin yrityselämässäkin. Datavirheitä syntyy väistämättä esimerkiksi ihmisen tai järjestelmän virheen toimesta. Virheillä voi olla liiketoiminnan kannalta ikäviä seurauksia, mutta niitä voidaan karsia automaatioilla. Tämä tutkielma on osoitus siitä.

Lähteet

[1] Alhyasat, Eiad Basher ja Al-Dalahmeh, Mahmoud: *Data warehouse success and strategic oriented business intelligence: a theoretical framework*. arXiv preprint arXiv:1307.7328, 2013.

[2] Ali, Abid, Anand, Karandeep Singh, Sen, Vijay ja Fries, Robert M: *Driving data backups with data source tagging*, 2009. US Patent 7,568,124.

[3] Amazon: *Simple Queue Service*. `https://aws.amazon.com/sqs/`, [Online].

[4] Baca, Murtha: *Introduction to metadata*. Getty Publications, 2008.

[5] Ballou, Donald P ja Tayi, Giri Kumar: *Enhancing data quality in data warehouse environments*. Communications of the ACM, 42(1):73–78, 1999.

[6] Barateiro, José ja Galhardas, Helena: *A survey of data quality tools*. Datenbank-Spektrum, 14(15-21):48, 2005.

[7] Berry, Michael J ja Linoff, Gordon: *Data mining techniques: for marketing, sales, and customer support*. John Wiley & Sons, Inc., 1997.

[8] Bilenko, Mikhail ja Mooney, Raymond J: *Adaptive duplicate detection using learnable string similarity measures*. Teoksessa *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, sivut 39–48. ACM, 2003.

[9] Eckerson, Wayne W: *Data quality and the bottom line*. TDWI Report, The Data Warehouse Institute, 2002.

[10] El-Sappagh, Shaker H Ali, Hendawi, Abdeltawab M Ahmed ja El Bastawissy, Ali Hamed: *A proposed model for data warehouse ETL processes*. Journal of King Saud University-Computer and Information Sciences, 23(2):91–104, 2011.

[11] Elmagarmid, Ahmed K, Ipeirotis, Panagiotis G ja Verykios, Vassilios S: *Duplicate record detection: A survey*. IEEE Transactions on knowledge and data engineering, 19(1), 2007.

[12] Fernández, Alberto, Río, Sara del, López, Victoria, Bawakid, Abdullah, Jesus, María J del, Benítez, José M ja Herrera, Francisco: *Big Data with Cloud Computing: an insight on the computing environment, MapReduce, and programming frameworks*. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 4(5):380–409, 2014.

[13] Gartner: *Gartner Survey of 1,400 CIOs Shows Transformation of IT Organisation is Accelerating*, January 2006. `http://www.gartner.com/newsroom/id/492238`, [Online; posted January 23, 2006].

[14] Guyon, Isabelle, Matic, Nada, Vapnik, Vladimir *et al.*: *Discovering Informative Patterns and Data Cleaning.*, 1996.

[15] Harvey, Andrew C: *Forecasting, structural time series models and the Kalman filter*. Cambridge university press, 1990.

[16] Helfert, Markus, Zellner, Gregor ja Sousa, Carlos: *Data quality problems and proactive data quality management in data-warehouse-systems.* Proceedings of BITWorld, 2002.

[17] Hernández, Mauricio A ja Stolfo, Salvatore J: *The merge/purge problem for large databases.* Teoksessa *ACM Sigmod Record*, nide 24, sivut 127–138. ACM, 1995.

[18] Hernández, Mauricio A. ja Stolfo, Salvatore J.: *Real-world Data is Dirty: Data Cleansing and The Merge/Purge Problem.* Data Mining and Knowledge Discovery, 2(1):9–37, Springer, 1998.

[19] Inmon, William H.: *Building the Data Warehouse.*, nide 4th ed. Wiley, 2005.

[20] ISO/IEC 25012:2008(E) Software product Quality Requirements and Evaluation — Data quality model, International Organization for Standardization, Switzerland, 2008.

[21] Jeff Sutherland, Ken Schwaber: *Scrum Guides.* `http://www.scrumguides.org`, [Online].

[22] Kimball, Ralph ja Ross, Margy: *The data warehouse toolkit: the complete guide to dimensional modeling*, nide 3rd Edition. John Wiley & Sons, 2002.

[23] Maletic, Jonathan I ja Marcus, Andrian: *Data Cleansing: Beyond Integrity Analysis.* Teoksessa *Proceeding, 2000 MIT Information Quality Conference*, sivut 200–209, 2000.

[24] Monge, Alvaro E.: *Matching algorithms within a duplicate detection system.* IEEE Data Eng. Bull., 23(4):14–20, 2000.

[25] Müller, Heiko ja Freytag, Johann Christph: *Problems, methods, and challenges in comprehensive data cleansing.* Professoren des Inst. Für Informatik, 2005.

[26] Oliveira, Paulo, Rodrigues, Fátima, Henriques, Pedro ja Galhardas, Helena: *A taxonomy of data quality problems.* Teoksessa *2nd Int. Workshop on Data and Information Quality*, sivut 219–233. Citeseer, 2005.

[27] Oracle: *Class HashMap.* `https://docs.oracle.com/javase/7/docs/api/java/util/HashMap.html`, [Online].

[28] Oracle: *StringBuilder API.* `https://docs.oracle.com/javase/7/docs/api/java/lang/StringBuilder.html`, [Online].

[29] POI, APACHE: *Apache POI - the Java API for Microsoft Documents.* `https://poi.apache.org/`, [Online].

[30] Rahm, Erhard ja Do, Hong Hai: *Data cleaning: Problems and current approaches.* IEEE Data Eng. Bull., 23(4):3–13, 2000.

[31] Redman, Thomas C: *The impact of poor data quality on the typical enterprise.* Communications of the ACM, 41(2):79–82, 1998.

[32] Robson, Colin: *Real World Research: A resource for social scientists and resources*, 2002.

[33] Runeson, Per ja Höst, Martin: *Guidelines for conducting and reporting case study research in software engineering.* 2008.

[34] SAP: *SAP.* `www.sap.com`, [Online].

[35] Seaman, CB: *Qualitative methods in empirical studies of software engineering.* Software Engineering, IEEE Transactions on, 1999.

[36] Shafranovich, Yakov: *Common format and MIME type for comma-separated values (CSV) files.* 2005.

[37] Stellman, Andrew ja Greene, Jennifer: *Applied software project management.* O'Reilly Media, Inc., 2005.

[38] Strong, Diane M, Lee, Yang W ja Wang, Richard Y: *Data quality in context.* Communications of the ACM, 40(5):103–110, 1997.

[39] Subramanian, Ashok, Smith, L Douglas, Nelson, Anthony C, Campbell, James F ja Bird, David A: *Strategic planning for data warehousing.* Information & management, 33(2):99–113, 1997.

[40] Tuomi, Jouni ja Sarajärvi, Anneli: *Laadullinen tutkimus ja sisällön analyysi: Uudistettu laitos*, nide 2017. Tammi.

[41] Umble, Elisabeth J, Haft, Ronald R ja Umble, M Michael: *Enterprise resource planning: Implementation procedures and critical success factors.* European journal of operational research, 146(2):241–257, 2003.

[42] Vassiliadis, Panos, Vagena, Zografoula, Skiadopoulos, Spiros, Karayannidis, Nikos ja Sellis, Timos: *ARKTOS: A tool for data cleaning and transformation in data warehouse environments.* IEEE Data Eng. Bull., 23(4):42–47, 2000.

[43] Wang, Richard Y ja Strong, Diane M: *Beyond accuracy: What data quality means to data consumers.* Journal of management information systems, 12(4):5–33, 1996.

[44] Watson, Hugh J ja Wixom, Barbara H: *The current state of business intelligence.* Computer, 40(9):96–99, 2007.

[45] Watt, Adrienne ja Eng, Nelson: *Database Design*, nide 2nd Edition. BCcampus Open Textbooks, 2012.

[46] Wirth, Rüdiger ja Hipp, Jochen: *CRISP-DM: Towards a standard process model for data mining.* Teoksessa *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, sivut 29–39. ACM, 2000.

[47] Xiao, Chuan, Wang, Wei, Lin, Xuemin, Yu, Jeffrey Xu ja Wang, Guoren: *Efficient similarity joins for near-duplicate detection.* ACM Transactions on Database Systems (TODS), 36(3):15, 2011.

A Liite 1: Haastattelukysymykset - Vaatimusmäärittely

1. Nimi
2. Kuinka kauan olet ollut Relexillä?
3. Mitä teet Relexillä?
4. Kenen vastuulla on validoida dataa Relexin päässä?
5. Mitä asioita tarkistat, kun historiatapahtumat on luettu sisään?
6. Minkälaisia datan laatuvirheitä on tullut vastaan?
7. Mistä laatuvirheet ovat johtuneet?
8. Mitä tehdään, kun virheitä havaitaan?
9. Onko laatuvirheitä havaittu liian myöhään?
10. Minkälaisia laatuvirheitä on havaittu liian myöhään?
11. Mitkä datan laatuseikat ovat asiakkaiden kannalta tärkeimpiä?
12. Voisitko näyttää, miten validoit dataa?
13. Tämänhetkisen validointityökalun Excel-raportin esittely. Palaute raportista?
14. Tuleeko vielä jotain muuta mieleen? Onko kysymyksiä? Palaute haastattelusta?

A Liite 2: Haastattelukysymykset - Asiakashankkeet

1. When did the project start?
2. Have you started validating the data you have in the environment? How are you validating the data?
3. What kind of master data do you have in the environment?
4. What kind of data quality problems have you had related to master data?
5. What kind of transaction data do you have in the environment?
6. What kind of data quality problems have you had related to transaction data?
7. Was some interface iterated or modified several times? If yes, why?
8. Show & explain tool's findings. Is validition tool useful?